Problem description

Background

Multitarget tracking is a key ingredient in collision avidance system for autonomous vehicles. Multi-frame tracking methods are commonly acknowledged as gold standards for multi-target tracking. The purpose of this master thesis is to develop a complete multi-frame tracking system for autonomous ships, based on sensor inputs from radar and the Automatic Identification System (AIS).

Proposed tasks

The following task are proposed for this thesis:

- Extend an integer-linear-programming (ILP) based tracking method with suitable algorithms for track initiation and track management
- Develop a framework for fusion between radar tracks and AIS tracks
- Develop alternatives to N-scan pruning in order to enhance the computational efficiency of the tracking method
- Implement the tracking system in Python and/or C++
- Test the tracking system on simulated data

Autosea

This thesis is associated with the AUTOSEA project, which is collaborative research project between NTNU, DNV GL, Kongsberg Maritime and Maritime Robotics, focused on achieving world-leading competence and knowledge in the design and verification of methods and systems for sensor fusion and collision avoidance for Autonomous Surface Vessels (ASVs). The project has access to supervision and physical test platforms through our industry partners.

Preface

This Master thesis is written at Norwegian University of Science and Technology (NTNU) as final work in the Engineering Cybernetics study program with specialisation in Robotics and Vessel control and was carried out during the spring of 2017.

I would like to thank the people that have made this thesis a reality. First of all, my supervisor Associate Professor Edmund F. Brekke for giving me freedom and trust to seek out my own solutions, and give guidance with constructive feedback. Next, a great thanks to my Co-Supervisors Ph.D. students Erik F. Wilthil and Andreas L. Flåten for helpful discussions and for giving me access to our computing server 'Syn'. I would also thank Paal Kristian for a being a great companion at the office.

This thesis would not have been completed without the patience and understanding from my girlfriend Elisabeth, thank you.

Erik Liland
Trondheim, June 2017

Abstract

To enable autonomous sea vessel's safe voyage, a real time situational awareness system is required. A tracking system incorporating both radar and Automatic Identification System (AIS) sensor data is preferred in maritime situations since radar and AIS have different strong and weak properties. A multi-frame multitarget tracking system based on radar measurements from own vessel, aided by AIS messages is developed. The system consist of two main parts, a logic based initialization algorithm and a Track Oriented Multi Hypothesis Tracker (TOMHT), both utilizing radar and AIS. This tracking system is demonstrated on simulated multitarget data with different tuning settings, external environment and AIS configurations.

The track loss improvement for for window sizes of 3–9 of all targets equipped with class A AIS in a cluttered environment with low Probability of detection (P_D) (50%) was 85–94%. The tracking time percentage was improved by 43% for small (N=3) multi-frame window sizes and 8–15% for larger (N=6–9) window sizes when comparing pure radar measurements with full class A AIS coverage for very low detection probability (50%) and high clutter density.

Sammendrag

Løysinga er 42.

Contents

List of Figures

List of Tables

Glossary

VHF The frequency range between 30 and 300 MHz.

Algorithm Step by step operations to be performed.

Clutter Noise in the form of false measurements where the amount is assumed Poisson distributed.

Coalescence Come together to form one whole.

COLREGS Convention on the International Regulations for Preventing Collisions at Sea.

Gate An area in which a track expects and approves new measurement to associate with itself.

Gross tonnage A measurement of a ship's overall internal volume.

Measurement A point in the measurement space where something is detected.

Measurement list A set of measurement which originate from the same scan.

Measurement noise Also called observation noise, which is noise that affects the accuracy of the measurement not the existence. White Gaussian with zero mean and covariance $\boldsymbol{R}$.

Nautical Mile Length used in maritime navigation. Equals 1 minute of latitude (1852 meters).

NMEA Communication protocol used between electronic maritime equipment, based on RS-422.

Python An open-source programming language.

Radar Acronym for Radio Detection And Ranging. A device that uses radio waves to measure distance and bearing to other objects.

RS-232 Serial single ended communication standard.

RS-422 Serial differential communication standard.

Score A measure of the goodness of a measurement-to-track association.

SOLAS The International Convention for the Safety of Life at Sea.

Solver A program that solves optimization problems.

System noise Also called process noise, which is noise in the model behaviour. This noise compensates for the uncertainty and non-modelled dynamics of the true system.

Target An actual object which the system is trying to track.

Track forest A forest of track hypothesis trees.

Track hypothesis A leaf node with its predecessors.

Track hypothesis tree An acyclic graph spanning from the root node of each target where each node is a track hypothesis.

Tracking The process of initiating, maintaining and terminating tracks from measurements.

Acronyms

P_D Probability of detection

AIS Automatic Identification System

ASV Autonomous Surface Vessel

BFS Breath First Search

CAS Collision Avoidance System

CDF Cumulative Distribution Function

CNLLR Cumulative Negative Logarithmic Likelihood Ratio (NLLR)

CPA Closest Point of Approach

CPHD Cardinalized Probability Hypothesis Density

CSTDMA Carrier Sense Time Division Multiple Access

DAG Directed Acyclic Graph

DFS Depth First Search

DNV GL Det Norske Veritas Germanischer Lloyd

FISST Finite Set Statistic

HOMHT Hypothesis Oriented Multi Hypothesis Tracker

ILP Integer Linear Programming

IMO International Maritime Organization

JPDAF Joint Probabilistic Data Association Filter

MHT Multi Hypothesis Tracking

MILP Mixed Integer Linear Programming

MMSI Mobile Maritime Safety Identity

MUNIN Maritime Unmanned Navigation through Intelligence in Networks

NIS Normalized Innovation Squared

NLLR Negative Logarithmic Likelihood Ratio

NM Nautical Mile

NNF Nearest Neighbour Filter

NTNU Norwegian University of Science and Technology

PDAF Probabilistic Data Association Filter

PDF Probability Density Function

PHD Probability Hypothesis Density

RADAR RAdio Detection And Ranging

RAM Random Access Memory

RFS Random Finite Set

RMS Root Mean Square

RMSD Root Mean Square Deviation

RPM Rotations Per Minute

SAR Synthetic Aperture Radar

SOTDMA Self Organized Time Division Multiple Access

SSD Solid State Storage

TDMA Time Division Multiple Access

TOMHT Track Oriented Multi Hypothesis Tracker

UTC Coordinated Universal Time

UTM Universal Transverse Mercator coordinate system

VHF Very High Frequency

VTS Vessel Traffic Service

Nomenclature

$\bar{\mathbf{x}}$	Predicted state
$\hat{\mathbf{x}}$	Filtered state
λ_ν	Poisson spatial density of the number of new measurements
λ_ϕ	Poisson spatial density of the number of false measurements
λ_{AIS}	Possion spatial density of the number of 'extraneous' AIS measurements
λ_{ex}	Total spatial density of the number of "extraneous" measurements
CNLLR	Cumulative Negative Log Likelihood Ratio
NLLR	Negative Log Likelihood Ratio
μ_a	Average true converted measurement bias
$\bar{\boldsymbol{P}}$	Predicted state covariance
$\hat{\boldsymbol{P}}$	Filtered state covariance
$\boldsymbol{\Phi}$	State transition matrix
$\boldsymbol{H}$	State observation matrix
$\boldsymbol{K}$	Kalman gain
$\boldsymbol{Q}$	Process noise covariance matrix

$\boldsymbol{R}$	Measurement covariance matrix
$\boldsymbol{S}$	Residual covariance
σ_r	Range measurement standard deviation
σ_θ	bearing measurement standard deviation
θ_m	Measured angle
$\hat{\mathbf{z}}$	Predicted measurement
$\boldsymbol{\tau}$	Binary vector where the selected track hypotheses are 1
$\tilde{\mathbf{x}}$	Measurement innovation
$\mathbf{v}$	Observation noise
$\mathbf{w}$	Process noise
$\mathbf{x}$	State vector
$\mathbf{z}$	Measurement
i	Measurement index
j	Target index
k	Time index
l	Hypothesis index
m_k	Number of measurements in scan k
ms	Millisecond
N	Number of scans to keep in track tree
r_m	Measured range
t	Time
t_k	Time at time index

Chapter 1

Introduction

1.1 Motivation

Automation- and control technology have throughout the history been a crucial part of relieving humans from for instance dangerous, exhaustive, repetitive or boring work. Examples of this is automation and robotics in production facilities, remotely operated vehicles for working and exploring the deep sea, disarming explosives and explore space. The level of self control varies from remotely controlled to self sensing and planning without human interaction.

The early motivation for automation was probably, and in many situations still are, to improve speed, quality and consistency, which all tends to lead to better economics. With a still decreasing threshold for automating processes, more focus is applied on easing the burden on people, either by combining robotics and humans in the same operation, or by fully automate the task. These jobs are typically repetitive, dangerous or both.

Although humans are capable of both self improving and easily adapting to new tasks, they will always have good and bad days, performing the same task slightly different or be bored and unfocused. These are all aspects that leads to inconsistency and errors, which may not be a problem in a production environment with quality inspections, though inconvenient, but can be fatal in critical applications.

There also exists several places where humans and automated system work together to exploit the strengths from both humans and machines, for instance in aviation where the pilots are always present in the cockpit, but the autopilot are flying the plane most of the time. This gives the pilots freedom from a very static and repetitive task where a

human error could have fatal consequences. This symbiosis is somewhat similar to the workload on the bridge of commercial vessels, where the autopilot is steering the ship most of the time, while the crew is setting the course.

For vessels that do very repetitive routes and jobs, like ferries and short domestic cargo transport, the mental fatigue on the crew can be an issue. Because of the need for crew in emergency situations, customer service and ship maintenance, larger ferries would still need crew if their navigation were to be automated. The vessels could, however, be controlled by an automated route planning- and Collision Avoidance System (CAS). The control system would never be tired, bored, intoxicated or distracted in the same ways as humans can. These are some aspects that make Autonomous Surface Vessels (ASVs) applicable for certain use cases.

The sensor and control system needed for safe automation of any vessel is large and complex, and requires several layers of fault barriers to prevent system errors from spreading in addition to the ability to self monitor its own performance. The control system would know its own position and desired position, it would have access to maps to make a route, a CAS to deviate from its planned route to act in accordance with the rules at sea (COLREGS) based on real-time situation information from the sensors on the vessel.

For ASVs to be a viable alternative to human guided ships, the potential savings must be more than marginal, and the control system must be at least as safe as a human operated vessel. The state-of-the-art is not at this point yet, but recent initiatives by large corporations in development in ASVs and the regulation of a dedicated test area for ASVs in Trondheimsfjorden in Norway are just two examples of the direction this technology is headed.

The worlds first autonomous ferry might be between Ravnkloa and Vestre kanalkai in Trondheim or between Breivik and Larvik in Porsgrunn. The first project is a collaboration between Norwegian University of Science and Technology (NTNU) and Det Norske Veritas Germanischer Lloyd (DNV GL), with the aim to develop a small autonomous battery powered passenger and bike-cycle ferry as an alternative to a bride over a canal. The second project is a partnership between YARA and Kongsberg Maritime, with the goal of having a domestic short range electric vessel fully autonomous by 2020.

Another indicator of the momentum autonomous surface vessels have is the Maritime Unmanned Navigation through Intelligence in Networks (MUNIN) project, which is a collaborate project between several European companies and research institutes, partially funded by the European Commission. The project aims at developing and verifying

concept of autonomous vessels with remote control from onshore control stations.

This work is focused on the sensor fusion which generates a real time data stream into the control system, enabling situational awareness and the foundation for predictive CAS like [1].

1.2 Previous work

This work is based on a pre-master project executed autumn 2016 [2]. In this project, it was shown that several off-the-shelf Integer Linear Programming (ILP) solvers was capable of solving the data association optimization problem in a single sensor Track Oriented Multi Hypothesis Tracker (TOMHT). It also showed that under good to moderate conditions, the performance return when increasing multi-scan window more than a relative low threshold, was very low.

1.3 Outline of the Thesis

Chapter 2 provides an introduction to the sensor systems used in this work, as well as some of the different flavours of tracking methods that exist. Chapter 3 gives a brief introduction to radar and Automatic Identification System (AIS) as systems, with focus on their pre-processing requirements prior to the MHT module. Chapter 4 presents an overview of the complete measurement-to-track system and an in-depth explanation of the fused radar and AIS TOMHT tracking system. Chapter 5 presents the different scenarios that are used in the performance evaluation of the tracker and the results of the simulated scenarios. A discussion of the results and evaluation of the performance with respect to safety at sea is presented in Chapter 6, followed by suggestion for future work in Section 6.1. The thesis is concluded in Chapter 7.

Chapter 2

Theoretical Background

2.1 Radar

2.1.1 Overview

RAdio Detection And Ranging (RADAR) is a detection technology that uses radio waves to observe stationary and moving objects. A transmitter sends out radio waves and a receiver is waiting for reflected echo's from objects, were the time the echo is delayed determines the distance to the object. The transmitter and receiver will in many situations be in the same location, and can be both stationary or mobile and have fixed or rotating orientation. Depending on frequency, a radar can observe solid objects like aircraft, ships, terrain, road vehicles and less solid objects like people and weather formations.

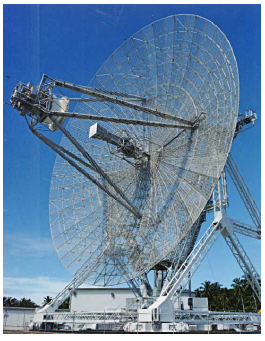

Figure 2.1: Fixed radar antenna

Figure 2.2: Rotating radar antenna

Figure 2.3: Maritime radar antenna

2.1.2 History

The fist implementation of an instrument that were able to detect the presence of distance metallic objects by radio waves was done by Christian Hülsmeyer in 1904. His invention did not measure the distance to objects, but whether there was an object in the direction of the instrument. The radar as we know it today was introduced in the mid to late 1930's, with world war two triggering research to improve the still immature technology to be used in military applications. After the war, the technology matured and were put in use in several civil applications, where air traffic control, maritime safety and weather monitoring is the most common.

2.1.3 Principles

The electromagnetic waves that a radar emits travels at the speed of light in air and vacuum. It reflects back when there is a change in the density of the medium it is travelling through, which is what happens when radio waves hit targets. Electrically conductive materials tend to be good reflectors, since they have a very different atomic density than air. On the other hand, materials with poor conductivity, and also some magnetic materials, tend to absorb radio waves. Like light, there are many ways an incoming radio wave can be reflected, primarily dependent on the geometry of the target. A corner with angles less than 90° will reflect the incoming radio waves directly back to the sender, and is a good thing on targets that want to be visible on a radar. This principle is the basis for radar reflectors commonly used to boost the radar signature on smaller vessels, see Figure 2.4. The opposite is used on targets that try to minimize their radar signature, and is the reason why stealth vessels and aircraft are tiled by flat areas.

Figure 2.5: USS Zumwalt

Figure 2.6: KNM Gnist

Figure 2.7: F177 Nighthawk

2.2 AIS

The Automatic Identification System (AIS) is a maritime safety and information system primarily designed for collision avoidance. AIS works by broadcasting messages on the Very High Frequency (VHF) band at irregular intervals with information on the vessels. AIS transceivers are required on international voyaging vessels over 300 gross tonnage, and on all passenger vessels. AIS signals are received at both vessels and shore stations for use in Vessel Traffic Service (VTS) stations, open tracking databases like www.marinetraffic.com, fleet-monitoring and search and rescue operations. Since the AIS messages contains position, course and speed, AIS tracks can be overlaid on a map in a chart plotter or on top of a radar image, giving the operator two sensors to verify each other.

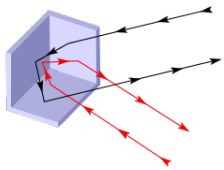

Figure 2.4: Corner reflector

2.2.1 History

AIS was designed and developed by technical committees in the International Maritime Organization (IMO). Its objective was to enhance vessels safety and efficiency by increasing their ability to see and identify other vessels. The main motivation for adopting AIS was its independence of humans in operation, since it automatically identifies other vessels and displays the information on the navigational system on the bridge. It also enables automatic calculation of Closest Point of Approach (CPA) and time until CPA, from which the navigation system could alarm the bridge of incoming traffic on dangerous course. This gives the navigator on the bridge more and better information for making decisions, but with the caveat that not all vessels have AIS. In the 2002 IMO SOLAS Agreement, it is required that vessels over 300 gross tonnage and all passenger vessels must be equipped with class A AIS transceivers. A simpler and cheaper AIS version named class B aimed at smaller vessels and yachts was published in 2006, followed by a large increase in the amount of non-commercial vessels equipped with AIS.

2.2.2 Messages

AIS broadcasts both static, dynamic and voyage information with varying intervals based on the vessels speed, status and on request from shore stations. Static, dynamic and voyage messages are listed in Table tables 2.1 to 2.3. When the AIS standard was developed, the peak traffic situations in the two most densely trafficked waterways, Singapore and

Dover Straits, were used to calculate the update frequency for the AIS system. Based on these two locations and a desire to keep the number of reports per minute below 2000, the dynamic information report intervals for class A and B was set as in Table 2.4 and 2.5 respectively [3]. Static information is transmitted every 6 minutes, and on request from VTS stations. AIS transceivers are utilizing two reserved VHF channels; AIS 1 — 87B (161.975MHz) and AIS 2 — 88B (162.025MHz) to improve robustness against interference. An important note is that AIS transceivers are alternating which channel they are transmitting on, which means that if a receiver is only listening on one channel, the effective update rate halves.

2.2.3 Class A

Class A AIS transceivers are designed for Self Organized Time Division Multiple Access (SOTDMA) transmission, which is a way of reserving transmission time slot for the next broadcast. SOTDMA is based on Time Division Multiple Access (TDMA), with an extension allowing for self organizing of time slots compared to TDMAs dedicated timing manager. This effectively gives class A AIS transmissions priority over Class B equipment which may not have SOTDMA. Class A transceivers are also required to have build-in display, minimum transmission power of 12.5W, ability to filter targets and communication interfaces like RS-232 and NMEA.

2.2.4 Class B

Class B AIS transceivers are designed to be simpler and cheaper than Class A transceivers, which is accomplished through less strict requirements for hardware and operation. Class B AIS transmits at lower power, usually 2W and transmits at larger time intervals than Class A, see Table 2.5. It is not required to have a build-in display and can use both Carrier Sense Time Division Multiple Access (CSTDMA) and SOTDMA for transmission. CSTDMA is a simpler approach to time division than SOTDMA since it only listen for a single time slot to be unused before it transmits.

Static AIS information	
MMSI	Maritime Mobile Service Identity
Call sign	Maritime radio (VHF) call sign
Name	Name of vessel
IMO Number	Vessel IMO number
Length and beam	
Location of positioning fixing antenna	
Height over keel	

Table 2.1: Static AIS information

Dynamic AIS information	
Position	In WGS84 frame
Position accuracy	Better or worse that 10 meter
Position time stamp	UTC in whole seconds
Course over ground (COG)	
Speed over ground (SOG)	
Heading	
Navigational status	
Rate of turn (ROT)	

Table 2.2: Dynamic AIS information

Voyage AIS information	
Draught	Depth in water
Hazardous cargo	Type
Destination	Name of place
Estimated time of arrival (ETA)	
Route plan / waypoints	
Number of persons on board	

Table 2.3: Voyage AIS information

Vessels status	Reporting Interval
Ship at anchor or moored and not moving faster than 3 knots	3 minutes
Ship at anchor or moored and moving faster than 3 knots	10 seconds
Ship 0–14 knots	10 seconds
Ship 0–14 knots and changing course	3.3 seconds
Ship 14–23 knots	6 seconds
Ship 14–23 knots and changing course	2 seconds
Ship > 23 knots	2 seconds
Ship > 23 knots and changing course	2 seconds

Table 2.4: Class A Reporting Intervals

Vessels status	Reporting Interval
Ship < 2 knots	3 minutes
Ship 2–14 knots	30 seconds
Ship 14–23 knots	15 seconds
Ship > 23 knots	5 seconds
Search and Rescue aircraft	10 seconds
Aids to navigation	3 minutes
AIS base station	10 seconds

Table 2.5: Class B Reporting Intervals

2.3 Tracking

2.3.1 Overview

Tracking, in this context, is the process of estimating the state of stationary and moving targets that are observed by a system without included association data. A key challenge is to know which measurements belong together over time, often referred to as the data association problem. An observation system can be a radar, sonar or any other sensor that, passively or actively, detects objects within an area or volume. Any observation system will be prone to noise, both in form of internal- and external noise from the environment. This noise will cause false measurements that the tracking system must take care of. These false measurements are often referred to as clutter.

In this work 'tracking algorithm' will be used to describe the main logic in a tracking method or approach, while 'tracking system' will be used on complete systems with everything around the main algorithm included. A tracking system can be defined as: *A system that process consecutive measurements from one or more observation system and associates measurements from the same target into tracks with initialization of new tracks and termination of dead tracks.* A track is a sequence of states associated with a subset of all measurements from the observation systems.

Tracking is a relative new field of study, driven by the military and aerospace industry and enabled by the development of microprocessors and computers from the 1960's. The applications ranges from sonar tracking of submarines from submarines and navy surface vessels to air traffic control and missile guidance. This historical background is likely the reason for most published papers using these types of applications as background for testing. In recent years, tracking people and vehicles from visual- and Synthetic Aperture Radar (SAR) imagery have also become a topic in the research community [4]–[6]. New applications areas like oceanography, autonomous vehicles and biomedical research have also found use of tracking [7]–[9].

There are several factors contributing to the challenge of good tracking; clutter, lower than unity Probability of detection (P_D), multiple detections of the same vessel and wakes. Clutter is a term for unwanted measurements or noise, which is inherent in every observation system. For a maritime radar, this can be caused by waves, rain, snow, birds or shore echo. A common assumption on clutter is to assume the amount being Poisson distributed, and spatially uniformly distributed. P_D is a measure of how persistent the target is in the measurements, and will vary much between different types of targets, primarily dependent on their size, construction material and shape. Multiple detections

of each target can occur when the target is large and have several distinct areas which reflects radar waves better than the rest of the vessel, for instance when the hull of a boat is made from fibreglass and the metal objects inside is reflecting. Wakes are reflection caused by the turbulent water behind a moving object, which can be a problem for both sonar and maritime- and air-radar.

2.3.2 Single-target tracking

The simplest approach to tracking is single-target tracking, where it is assumed that it is only one target in the surveillance area, and any other measurement is regarded as either extra measurements of the target or clutter.

Nearest Neighbour Filter (NNF)

The simplest single-target tracking algorithm is the NNF, where the closest measurement is always selected [10]. This approach is very vulnerable to clutter and dense multitarget scenarios.

Probabilistic Data Association Filter (PDAF)

The most popular single-frame single-target tracking algorithm is PDAF, which calculates probabilities of all target to measurement association for all measurement inside its gate. A gate is an ellipse (2D) or ellipsoid (3D) which outlines the confidence area / volume for a predicted state and covariance for a given confidence value. The state update is then based on a weighted sum of measurement innovations where the weightings are the probabilities for their respective innovations. PDAF, like most tracking algorithms, assumes that at most one measurement is originating from the true target at each scan. It does not have a build-in initialization routine, and is dependent on an external initialization algorithm. One major drawback with PDAF if using it to track multiple targets is its vulnerability to track coalescence, which is when two tracks are merged into one 'average' in between them [11].

2.3.3 Multitarget tracking

A more generalized approach assuming that there can be any number of targets is called multitarget tracking, where the problem expands to jointly estimate several targets trajectories.

While a large number of tracking techniques have been developed, the three most used are Joint Probabilistic Data Association Filter (JPDAF), Multi Hypothesis Tracking (MHT) and the Random Finite Set (RFS) paradigm [9]. Compared to MHT and JPDAF, RFS is a relatively new approach to tracking, and has not reached the same maturity as MHT and JPDAF. MHT and JPDAF also differs from RFS in that they both do data association and filtering, whereas RFS directly seeks both optimal and suboptimal estimates of the multitarget state [9].

JPDAF

JPDAF is a multitarget expansion of PDAF which is a single-target tracking technique. JPDAF calculates joint posteriori association probabilities for every target in every scan. Each targets probability is a weighted sum over an exponential number of association hypotheses, where the weights are the key difference between PDAF and JPDAF. Like MHT, JPDAF suffers from high computational cost. However, various approximations have been proposed, and an efficient implementation exist and is patented by QinetiQ [12]. JPDAF, like its single target brother JPDAF also suffer from track coalescence.

MHT

MHT is a decision logic which generates and maintains alternative hypotheses when new measurements are received within the gate. By making several possible hypotheses, the decision on which measurement to choose can be propagated into the future when more information is available. MHT is a multi frame method, meaning it has the ability to utilize multiple scans to make better decisions. Each hypothesis is given a score based on a likelihood ratio as a reflection of how well the measurement fits the model, which are accumulated to evaluate the combinations of consecutive measurements.

In contrast to the PDAF and JPDAF methods which suffers from track coalescence [11], [13], MHT methods split when in doubt. The idea of using multiple hypotheses was first introduced by [14], but the first complete algorithm was presented in [15], where a Hypothesis Oriented Multi Hypothesis Tracker (HOMHT) was developed. Following this, a TOMHT was proposed in [16] and the score function for MHT was later deduced and discussed in [17] since no explicit track-score function were given in [16]. MHT is, in the same way as PDAF/JPDAF, developed under the main assumption that each target gives rise to maximally one measurement (possibly zero if P_D less than 1). The MHT approach to tracking and data association has been for a long time dismissed by many because of its computationally large cost. The dramatic increase in computational capability from

the 1980's to the late 2010's has lead to a new spring for MHT, with an increasing interest for use in tracking system. In 2004 Blackman stated that "Multiple hypothesis tracking is generally accepted as the preferred method for solving the data association problem in modern multiple target tracking system" [18]. Already in 2001 did Blackman publish a demonstration that MHT is capable of real-time demands [19].

MHT comes in two variations, HOMHT and TOMHT. They differ in their approach to arrange the measurements into hypotheses in that HOMHT builds hypotheses that are different ways of organizing the measurements into tracks, while TOMHT maintains already existing tracks and a hypothesis is only a possible track for a single target and not all targets.

RFS

RFS is a family of Bayesian methods and filters that is based on representing a multitarget state as a finite set of single target states. This leads to a more compact formulation of multitarget tracking, where the entire tracking problem can be expressed in terms of Bayes rule and the Chapman-Kolmogorov integral. This is known as the multitarget Bayes filter.

The RFS approach to multi sensor multitarget tracking was done by Mahler in 1994, which lay the foundation for the development of Finite Set Statistic (FISST). One popular filter based on RFS is the Probability Hypothesis Density (PHD) filter. The PHD filter is an approximation of the multitarget Bayes filter derived by Mahler using FISST [9]. The PHD filter normally estimates the number of targets and then selects the same numbers highest probable tracks. One large drawback with the PHD filter is its assumption that the predicted multitarget RFS is Poisson distributed. This assumption is relaxed in the Cardinalized Probability Hypothesis Density (CPHD) filter where the prior and predicted multitarget densities are independently and identically distributed cluster processes [20].

Chapter 3

Radar and AIS preprocessing

The process from raw radar and AIS data to target tracks is made up from several processing steps. The aim of this chapter is to give the reader a basic understanding of these steps and their challenges.

3.1 Radar preprocessing

Rotating maritime radars (Figure 2.3) are wide and short, giving them a tall and narrow beam. A ping, transmit and receive sequence is carried out for each antenna rotation angle in the radars scan resolution. This gives reflections as signal level in spokes described by polar coordinates, rotation angle and distance. Each spoke has a width determined by the design of the antenna, primarily the width of the antenna, and a number of cells dictated by the discretization and sampling interval of each spoke. The spokes is then run through a detection algorithm, which is filtering the received signal according to detection setting. The detection algorithm is often built in to the radar system, with both fixed and user adjustable detection parameters.

When displayed on a screen in a vessel, the output from the detection step is viewed and interpreted by the operators. In an automated scenario with autonomous vessels, the next step would be to transform the detections from polar vessel body frame to for instance a Cartesian world fixed local frame. Which frame to convert to is a design choice, and can be dependent on use-case, interconnected systems and performance requirements. This transformation is strongly dependent on knowing the position and attitude of own vessel at each spoke sampling time, which is fed from the vessel's navigation

system.

With all the spoke resolution cells converted to a world-fixed Cartesian coordinate system, it is desirable to remove land reflections, if any such are present. This step is dependent on highly detailed digital maps of the area in question, which are commercially available for most of the world. Since maps have both offsets and inaccuracies to some extent, a cleaner land masking can be accomplished by dilating the coastline. This is in many situations acceptable since the vessels will never be that close to shore, and any targets masked away is in a region out of interest.

The last step in the radar processing chain is to convert a point cloud into measurements, as one target will in most cases fill multiple resolution cells and therefore it does not yield good result to send all cells with detection forward as measurements. This clustering of the detections is also desirable due to the assumption that each target maximum generates one measurements. This leads to clustering algorithms that assumes that detections closely spaced are originating from the same target, and thus should be one measurement. There are many clustering algorithms available to solve this problem, some build graphs with vertices between neighbouring detections given a neighbour criterion, some estimate the number of clusters and optimizing the detections into this number of clusters [21]–[23]. When a set of detections are clustered together, their respective measurement is calculated as the centroid of the polygon made from the detections, which would be weighted by their signal strength if available. These measurements are sent to the tracking module.

3.1.1 Frame conversion

The radar measurements is by nature in polar frame, and the target motion model is best described in a Cartesian frame. The most usual solution is to convert the radar measurements to a Cartesian frame, and to avoid biased and optimistic covariances of the converted measurements, a procedure which compensates for these errors, (3.1) and (3.2), should be used in stead of the standard conversion (3.3) and (3.4) [24].

$$\begin{aligned} \begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} r_m \cos\theta_m \\ r_m \sin\theta_m \end{bmatrix} - \mu_a \\ \mu_a \triangleq \begin{bmatrix} E[\tilde{x}|r_m, \theta_m] \\ E[\tilde{y}|r_m, \theta_m] \end{bmatrix} &= \begin{bmatrix} r_m \cos\theta_m (e^{-\sigma_\theta^2}) - e^{-\sigma_\theta^2/2} \\ r_m \sin\theta_m (e^{-\sigma_\theta^2}) - e^{-\sigma_\theta^2/2} \end{bmatrix} \end{aligned} \tag{3.1}$$

$$
\begin{aligned}
R_a^{11} &\triangleq r_m^2 e^{-2\sigma_\theta^2}[\cos^2\theta_m(\cosh 2\sigma_\theta^2 - \cosh\sigma_\theta^2) + \sin^2\theta_m(\sinh 2\sigma_\theta^2 - \sinh\sigma_\theta^2)] \\
&\quad + \sigma_r^2 e^{-2\sigma_\theta^2}[\cos^2\theta_m(2\cosh 2\sigma_\theta^2 - \cosh\sigma_\theta^2) + \sin^2\theta_m(2\sinh 2\sigma_\theta^2 - \sinh\sigma_\theta^2)] \\
R_a^{22} &\triangleq r_m^2 e^{-2\sigma_\theta^2}[\sin^2\theta_m(\cosh 2\sigma_\theta^2 - \cosh\sigma_\theta^2) + \cos^2\theta_m(\sinh 2\sigma_\theta^2 - \sinh\sigma_\theta^2)] \\
&\quad + \sigma_r^2 e^{-2\sigma_\theta^2}[\sin^2\theta_m(2\cosh 2\sigma_\theta^2 - \cosh\sigma_\theta^2) + \cos^2\theta_m(2\sinh 2\sigma_\theta^2 - \sinh\sigma_\theta^2)] \\
R_a^{12} &\triangleq \sin\theta_m\cos\theta_m e^{-4\sigma_\theta^2}[\sigma_r^2 + (r_m^2 + \sigma_r^2)(1 - e^{\sigma_\theta^2})]
\end{aligned}
\tag{3.2}
$$

$$
x = r_m\cos\theta_m \qquad y = r_m\sin\theta_m \tag{3.3}
$$

$$
\begin{aligned}
R_L^{11} &\triangleq r_m^2\sigma_\theta^2\sin^2\theta_m + \sigma_r^2\cos^2\theta_m \\
R_L^{22} &\triangleq r_m^2\sigma_\theta^2\cos^2\theta_m + \sigma_r^2\sin^2\theta_m \\
R_L^{12} &\triangleq (\sigma_r^2 - r_m^2\sigma_\theta^2)\sin\theta_m\cos\theta_m
\end{aligned}
\tag{3.4}
$$

r_m = measured range

θ_m = measured bearing

σ_r = range measurement standard deviation

σ_θ = bearing measurement standard deviation

$\boldsymbol{R_a}$ = average true converted measurement covariance

$\boldsymbol{R_L}$ = linearised converted measurement covariance

3.2 AIS preprocessing

AIS does not suffer from the association uncertainty, clutter and low accuracy like radar measurement. It does, however, have some issues caused by suboptimal or erroneous transmitter implementation, transmission collision caused by TDMA leading to ID (Mobile Maritime Safety Identity (MMSI)) swaps, and delayed messages leading to out of order reception. In order to remove most of these errors, it is desirable to filter the incoming AIS messages before sending them to the tracking module.

3.2.1 Out-of-order filtering

All AIS messages are stamped with the Coordinated Universal Time (UTC) of transmission, and 'frequently arrive out-of-order' [25], illustrated in Figure 3.1 [25] (with permission).

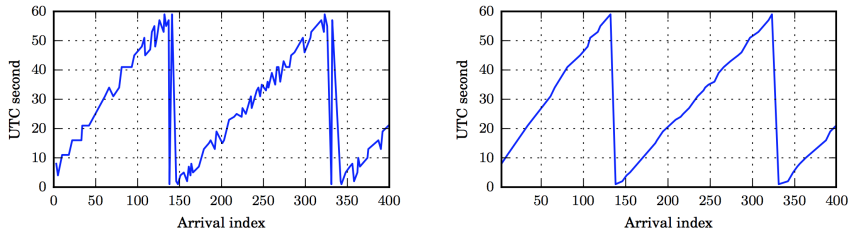

Figure 3.1: Unfiltered and filtered AIS arrival time

One of the simplest ways of remedying this issue is to discard all messages with older timestamps that the current newest for each MMSI. This will lead to a loss of data, which will lead to a slower AIS update period for the tracking module.

3.2.2 ID swap filtering

According to [26], 2% of the received AIS messages in a data-mining study contained erroneous MMSI. One of the errors were that many vessels transmitted messages with the same MMSI (11930446). This is the default MMSI on equipment from a specific manufacturer. Another example from the same study is two vessels which swapped IDs for a moment when they were passing, with a recovery after about 15 minutes. The latest example could be caused by simultaneous transmission or reflections, but the cause is not examined in the paper. Although much more rare that the out-of-order reception, ID swaps can occur and should be monitored by i.e. a logic testing AIS measurement innovations.

To remedy this issue, a simple test logic can be incorporated to check for obvious faults like sudden large position change and known default IDs. When a message and MMSI is categorized as bad, it would be held back from the tracking module.

Chapter 4

MHT Module

To create a complete tracking *system*, rather than a tracking *algorithm*, it is often necessary to complement the main algorithm with support modules. The system, or module if it is a part of a bigger system, presented here is an extension of the pre master project [2]. The aim of this chapter is to provide a complete walkthrough of the the track oriented MHT system developed in this thesis. This MHT module is based on both radar and AIS data from sensors mounted on own vessel. Since the radar is one of the most trustworthy sensors on board any vessel is this tracking module based on radar measurement primarily, with the AIS as an aiding system. This approach guided some of the design choices made throughout the development of the module.

The motion model which is used throughout the entire tracking system when predicting and filtering target behaviour is presented first. Next follows an overview of the algorithm used to initiate new tracks into the MHT algorithm, followed by the entire MHT tracking algorithm with all its sub-routines.

It will be assumed throughout this thesis that radar data is processed, as outlined in Section 3.1, into a set of points. These points is referred to as radar measurements.

4.1 Motion Model

4.1.1 Reference frame

A local Cartesian NED-frame, like Universal Transverse Mercator coordinate system (UTM) will be used throughout this thesis, with the assumption than all input sensors

are transformed to this frame (see Section 3.1.1). This local projection from a geodetic coordinate system to a Cartesian coordinate system is acceptable as long as the the area the system is working on is within one grid. A global geodetic frame, like WGS84 would be preferable in situations where the system tracks object over world-scale lengths but would yield non-linear equations of motion.

4.1.2 Constant velocity model

A target's state (4.1) is modelled with position and velocity in a 2D Cartesian frame where the positive x-axis is pointing east and the positive y-axis is pointing north. The two latest states are the velocities in their respective direction.

$$\mathbf{x} = \begin{bmatrix} x & y & \dot{x} & \dot{y} \end{bmatrix}^T \tag{4.1}$$

Since modelling the behaviour of any ship under unknown command is next to impossible, a common assumption in tracking theory is that every target will continue on as usual, more precisely that their velocity is constant. Although simple, this model captures the essence of most vessels at sea, and it should be noted that both maritime training [27] and regulation [28] dictates that vessels should hold steady course and change course in clear decisive turns. This model is also very common in tracking applications and is used in [6], [9], [15], [25], [29]–[31] among others. To give room in our model for manoeuvring, process noise is introduced with covariance set according to the assumed manoeuvring capabilities of the vessels. This could be set as a fixed value for all targets, as done in this work, or estimated based on the history of the track or AIS information. This behaviour can be modelled as a linear time invariant system with time evolution (4.2), measurement model (4.3), transition and observation matrices (4.4), and system and measurement noise matrices (4.5).

$$\mathbf{x}_{k+1} = \boldsymbol{\Phi}\mathbf{x}_k + \mathbf{w}_k \qquad \mathbf{w} \sim \mathcal{N}(0; \boldsymbol{Q}) \tag{4.2}$$

$$\mathbf{z}_{k+1} = \boldsymbol{H}\mathbf{x}_k + \mathbf{v}_k \qquad \mathbf{v} \sim \mathcal{N}(0; \boldsymbol{R}) \tag{4.3}$$

$$\boldsymbol{\Phi} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{4.4}$$

$$\boldsymbol{Q} = \sigma_v^2 \begin{bmatrix} \frac{T^3}{3} & 0 & \frac{T^2}{2} & 0 \\ 0 & \frac{T^3}{3} & 0 & \frac{T^2}{2} \\ \frac{T^2}{2} & 0 & T & 0 \\ 0 & \frac{T^2}{2} & 0 & T \end{bmatrix} \quad \boldsymbol{R} = \sigma_m^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.5}$$

$\boldsymbol{\Phi} =$ state transition matrix

$\boldsymbol{H} =$ state observation matrix

$\boldsymbol{Q} =$ system covariance matrix

$\mathbf{w} =$ system noise

$\mathbf{v} =$ measurement noise

$\mathbf{z} =$ measurement vector

$k =$ time index

$T =$ time step

4.2 Track Initiation

In comparison with HOMHT, which treats every measurement as a potential new track. TOMHT does not have any built-in initialization of tracks since it only maintains already existing tracks with track splitting and measurement-to-track association for every scan. To remedy this lack, we need an algorithm that can find consistent and predictable patterns in an assumed uniformly distributed measurement space of clutter.

In this work, new tracks are initiated with 2/2 & m/n logic [9] on the unused measurements after each MHT iteration. As the name of the method indicates, this is a two step verification, where the first act as a rough filter and the second as a fine filter. As a common in most tracking systems, the radar clutter is assumed uniformly distributed in the measurement area.

The flow of the method is illustrated in Figure 4.1, and for better clarity, the algorithm is explained from the last step to the first step, since this is the sequence a newly started initiation algorithm will perform its operations.

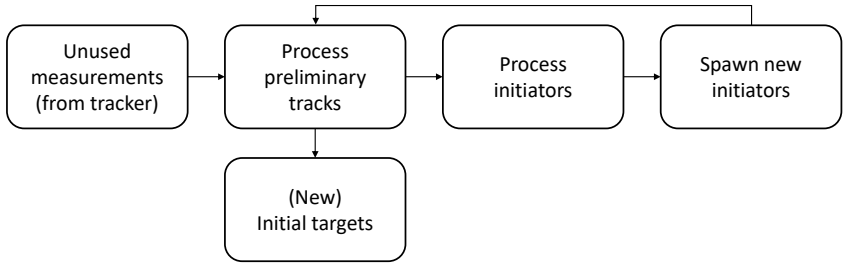

Figure 4.1: 2/2&m/n flowchart

4.2.1 Spawn new initiators

All measurement unused by the 'Process preliminary tracks' and 'Process initiators' steps will be the basis for new *initiators*. An initiator is a measurement that awaits its match in the next scan. The idea is that uniformly distributed clutter will not (often) reappear at approximately the same location two times in a row, effectively filtering out most of the clutter.

4.2.2 Process initiators

When the next scan arrives, all the unused measurements from the 'Process preliminary tracks' step will be used as candidates in this step. Since an initiator is only a position and not a full state with velocity, all directions are equally likely, and the only design parameter in this step is maximum speed of targets to be tracked. This parameter sets an outer limit on the circle acting as a gate for the second and confirming measurement. When matching initiators with a second measurement, we want to select the closest measurement, making the assumption that the two consecutive measurements are the most likely to belong together. In a single target scenario, where this would be to calculate the distance to all the alternative measurements and select the lowest, the association is already done. While in a multitarget scenario, we *could* select the closest measurement to any initiator, but we would have to do this one initiator at a time. This would lead to different results depending on the arrangement of the initiators in the programming of this method. A different approach would be to calculate all the different distances for any possible combination of initiators and measurements, sort the list, and assign the distances from the shortest to the longest possible distances. This approach would not be influenced by randomness like the arrangement of the initiators in a programming

language, but would not necessarily give the global optimal association regarding the how many initiators that are assigned measurements and their respective distances.

Since we can have situations like exemplified in Figure 4.2, where two initiators have the same measurement inside their gates, and one of them have a second measurement inside its gate, we need to take the global consequence of any assignment into consideration. If using method 1; to sequentially select the best, we have two possible outcomes. When starting with initiator 1, this initiator would be associated with measurement 2, and initiator 2 would not be associated with any measurements. On the other hand, starting with initiator 2 would lead to this initiator being associated with measurement 2, and initiator 1 would be associated with measurement 1. This randomness in outcome based on which initiator the algorithm starts with is clearly not a desired property. If using using method 2; to sequentially select the globally shortest distance, we would first associate initiator 1 with measurement 2, and there would not be any measurements left for initiator 2, leaving this empty.

A third option is to formulate the problem as a global combinatorial problem, and use an 'off-the-shelf' solution to solve the problem. We have essentially a matrix with initiators along one axis and measurements along the second axis and the distance between them in their intersections, as in (4.6) for our example.

$$\begin{array}{c} \\ I_1 \\ I_2 \end{array} \begin{array}{c} \begin{array}{ccc} M_1 & M_2 & M_3 \end{array} \\ \begin{bmatrix} 3 & 1 & 5 \\ 7 & 2 & 6 \end{bmatrix} \end{array} \tag{4.6}$$

The values above the threshold set by the maximum speed multiplied with the time period between the radar scans can be set to infinity to symbolise that this combination is not possible, see (4.7) where the gate threshold is 4.

$$\begin{array}{c} \\ I_1 \\ I_2 \end{array} \begin{array}{c} \begin{array}{ccc} M_1 & M_2 & M_3 \end{array} \\ \begin{bmatrix} 3 & 1 & \infty \\ \infty & 2 & \infty \end{bmatrix} \end{array} \tag{4.7}$$

If we remove the columns with only infinity, we are removing measurements that cannot be associated under any circumstances, thus reducing the size of the problem, see (4.8). With this pre processing, we want to assign each row to a column so that the sum of the

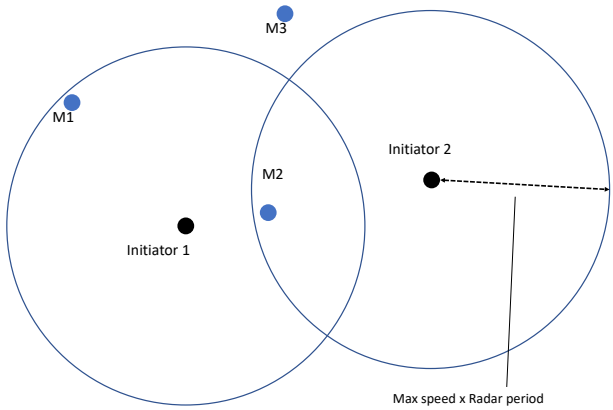

Figure 4.2: Initiator gating example

selected intersections are minimal.

$$\begin{array}{c} \\ I_1 \\ I_2 \end{array} \begin{array}{c} \begin{array}{cc} M_1 & M_2 \end{array} \\ \begin{bmatrix} 3 & 1 \\ \infty & 2 \end{bmatrix} \end{array} \tag{4.8}$$

We now have formulated our problem in a way that it can be solved by the 'Hungarian'[1] algorithm [32], which will give us the association $I_1 \rightarrow M_1$ and $I_2 \rightarrow M_2$. From the associations, a full state and new preliminary track is created with the latest measurement as position and velocity calculated based on the position difference divided by the time difference between the measurements. A preliminary track contains a state, initial covariance and counters of number of checks and passed checks. The initial covariance is a design variable and would be set according to the measurement and process noise.

4.2.3 Process preliminary tracks

When a new set of unused measurements arrive from the tracker, all the preliminary tracks are predicted to the time of the measurements. We now have the same association challenge between the predicted states and the measurements as with the initiators and measurements. Since we now have a full state and covariance for every preliminary track, we calculate the Normalized Innovation Squared (NIS) for every combination of preliminary tracks and measurements, and selects the best combination. The preliminary

[1] also known as the Munkres or Kuhn-Munkres algorithm

tracks that are associated with a new measurement, their passed counter is incremented with one, while all preliminary tracks' checks counter is incremented with one.

For preliminary tracks that have enough passed measurements, a new initial target is sent to the tracker. All preliminary tracks with check counter above the threshold is categorized as dead and deleted.

4.2.4 AIS aiding

Since we might have AIS data available, it would be desirable to use this to improve the time and reliability of the initialization. In the same way as unused radar measurements are the input to the initialization procedure, we can use the unused AIS measurements to skip the first 2/2 filtering and create a preliminary track for each AIS measurement. This way we are giving the AIS measurements a little more 'weight', but as this tracking system is based on AIS *aiding* we still want the m/n filtering to be done with radar measurements. To avoid creating duplicate preliminary tracks for the same MMSI over time, only unused AIS measurements with a MMSI not in the preliminary tracks are created. These choices are highly design specific and many other approaches is possible.

4.3 MHT Overview

The aim of this section is to outline the major steps in the MHT module and the flow of data and decisions. Figure 4.4 shows the main steps that the module perform at each iteration / radar scan. The MHT algorithm is working on a set of Directed Acyclic Graphs (DAGs) or tree structures, often called a forest. The forest contains as many trees as targets the algorithm is tracking. Each tree consist of a root node and a set of child nodes, where each row represent a scan or discrete time. The leaf nodes are referred to as track hypotheses since leaf nodes represents itself and its parents.

When new AIS and radar measurements are received, all leaf nodes are predicted forward to the time of the radar measurements. The radar measurements are then gated for each leaf node, and new pure radar hypotheses are generated for radar measurements within the gate. Each leaf node are then predicted to the time of the AIS measurements, and are gated at their time. AIS measurements inside the gate are then predicted to the time of the radar measurements where the radar measurements are gated based on each of the filtered AIS measurements. For each gated radar measurements give rise to fused hypotheses and from each AIS measurements without any radar measurement inside its gate a pure AIS hypothesis is created. Each new hypothesis is then given a score, which

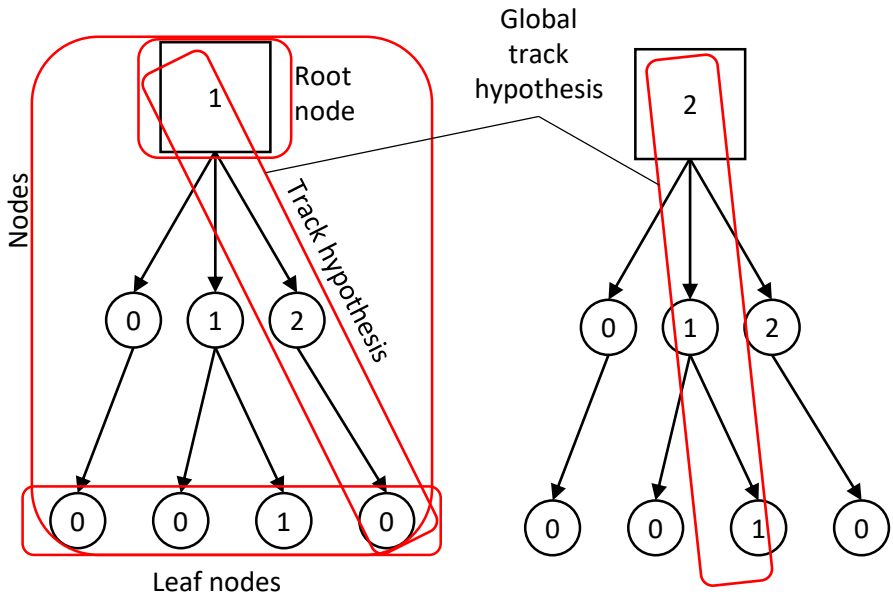

Figure 4.3: Hypothesis tree

is the cumulative score of the parent node score and the new node's score. The target trees are then clustered according to which trees that shares measurements, whereon clusters with only one tree has the option of removing / merging similar hypotheses to reduce the size of the tree. For each cluster, the cluster-wise globally best association combination is selected using ILP. Then, for each selected hypothesis the parent N steps above becomes the new root of that tree, and the unused children to the previous root node are removed. Next, targets whose best hypothesis have a score below the threshold is terminated, followed by the initialization of new targets from the initiator module.

The algorithm described in the three following sections are repeated for every leaf node in the forest. To avoid an extensive use of node- and measurement indexing the procedure is explained for one leaf node, with this node referred to as c—node, and is repeated for all leaf nodes in the forest. The c is only a symbol for indexing and referencing the chosen leaf node relative to predicted and filtered states.

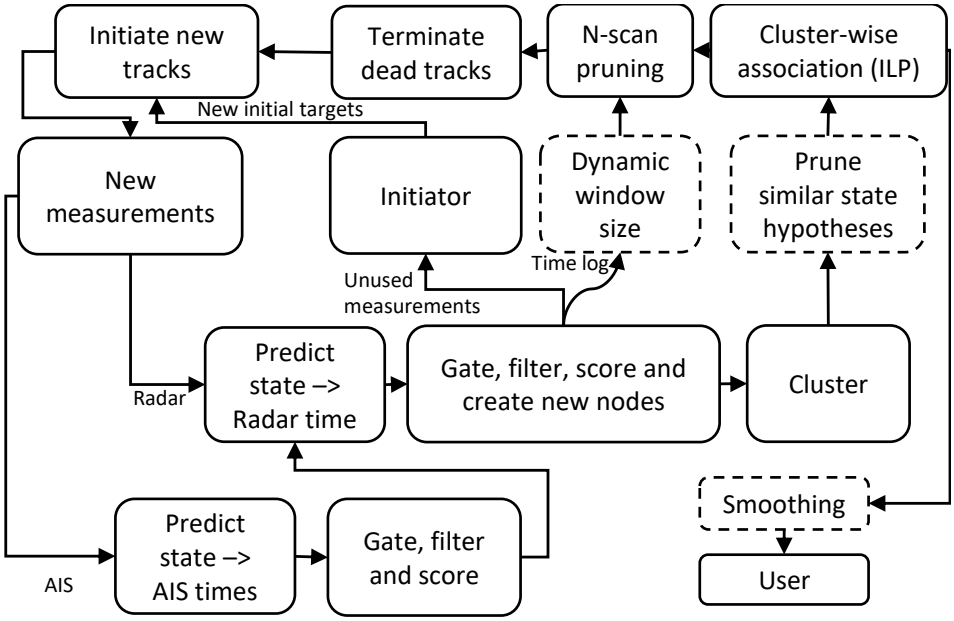

Figure 4.4: Algorithm flowchart

To illustrate how this MHT system works we can look at an example of a target when it is turning. Figure 4.5 shows a single target which started in the lower left side doing a hard port turn followed by a straight course. The purple dots is the true track, the black dot is the stating position, the solid line is the selected/best hypothesis and the dotted lines are all the other hypotheses. As the target is turning, the zero hypotheses will continue in a straight line to allow for the possibility that there is a missed detection at that time. Most of the zero hypotheses are also splitting after their first and second scan upon creation, which can be seen as sharp angles between dotted lines moving outwards in the turn and dotted lines representing hypotheses which have re-connected with the true target.

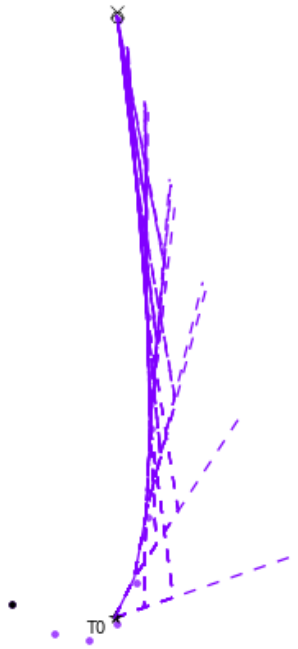

Figure 4.5: Hypotheses when turning

4.4 Process radar measurements

4.4.1 Predict to radar time

To compare new radar measurements with existing hypotheses, we should predict their states to the same time as the radar measurements, which can be done with the Kalman filter 'time update' equation (4.9) and the motion model from Section 4.1. The residual covariance (4.10), which is a part of the 'measurement update' sequence of a Kalman filter are also calculated as the residual covariance is needed in the gating and these matrices are not dependent on the measurement residual.

$$\begin{aligned} \bar{\mathbf{x}}_1 &= \boldsymbol{\Phi}(\Delta T_R)\mathbf{x}_c \\ \bar{\boldsymbol{P}}_1 &= \boldsymbol{\Phi}(\Delta T_R)\boldsymbol{P}_c\boldsymbol{\Phi}^T(\Delta T_R) + \boldsymbol{Q}(\Delta T_R) \end{aligned} \tag{4.9}$$

$$\boldsymbol{S}_1 = \boldsymbol{H}\bar{\boldsymbol{P}}_1\boldsymbol{H}^T + \boldsymbol{R}_{Radar} \tag{4.10}$$

$$\begin{aligned}
\bar{\mathbf{x}} &= \text{predicted state} \\
\mathbf{x}_c &= \text{origin node state} \\
\bar{\boldsymbol{P}} &= \text{predicted state covariance} \\
\boldsymbol{P}_c &= \text{origin node state covariance} \\
\Delta T_R &= \text{Radar time period}
\end{aligned}$$

4.4.2 Gate

To limit the number of hypotheses the leaf node have to create, the measurements are gated based on the leaf node's predicted covariance and a set confidence value. The size of the gate (Figure 4.6) will reflect how insecure the prediction is, which is a function of how many detections and missed detections the leaf node and its parents have had. The gate (4.11) is defined as NIS less than a threshold set by the inverse χ^2–distribution Cumulative Distribution Function (CDF) with as many degrees of freedom as the measurement, two degrees of freedom for a maritime radar, and a set confidence value. A set of confidence levels and belonging χ^2 CDF values are listed in Table 4.1. The measurement residual and NIS is calculated for each measurement, and the measurements that does not pass the test are discarded.

$$\begin{aligned}
\tilde{\mathbf{z}} &= \mathbf{z} - \boldsymbol{H}\bar{\mathbf{x}}_1 \\
NIS &= \tilde{\mathbf{z}}^T \boldsymbol{S}^{-1} \tilde{\mathbf{z}} \leq \eta^2
\end{aligned} \tag{4.11}$$

$$\begin{aligned}
\tilde{\mathbf{z}} &= \text{Measurement residual} \\
\eta^2 &= \text{Gate size}
\end{aligned}$$

4.4.3 Filter, score and create new nodes

The scoring used in this tracking system is based on a dimensionless score function by Bar-Shalom [17]. His paper discusses the issue of scoring measurement-to-track associations and comparing scores based on different numbers of measurement and measurement dimensions. He proposes a dimensionless *likelihood ratio*, which is the Probability

Confidence	70%	80%	90%	95%	97.5%	99%	99.5%
$\eta^2_{df=2}$	2.41	3.22	4.61	5.99	7.38	9.21	10.60

Table 4.1: Inverse χ^2 CDF for two degrees of freedom

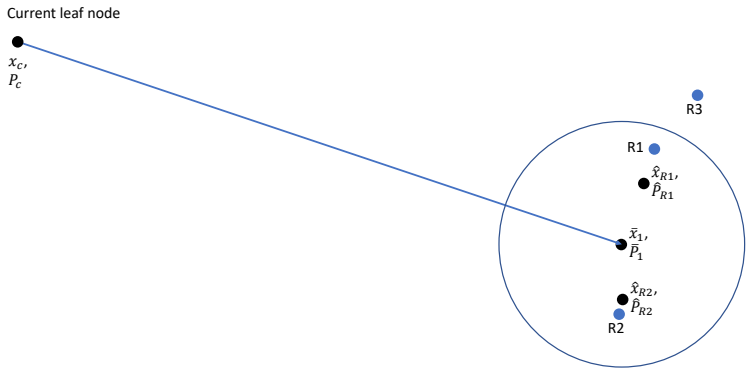

Figure 4.6: Gating radar measurements at radar time

Density Function (PDF) of a measurement having originating from the track, divided by the PDF of it not originating from the track. The outcome of not originating from the track is the union of the measurement being a clutter measurement and originating from a new target. The clutter and new target densities are both assumed Poisson distributed with λ_ν being the new target density and λ_ϕ the clutter density. The total extraneous measurement density is then $\lambda_{ex} = \lambda_\nu + \lambda_\phi$. For numerical

For each pure radar track hypothesis, a Negative Logarithmic Likelihood Ratio (NLLR) (4.12) is calculate, which is the negative natural logarithm of the score. The hypothesis is then scored with a Cumulative NLLR (CNLLR), which is the sum of its parent CNLLR and its own NLLR.

$$\text{NLLR}_{Radar} = \frac{1}{2} NIS + \ln \frac{\lambda_{ex} |2\pi S|^{1/2}}{P_D} \tag{4.12}$$

$$\text{CNLLR} \triangleq \sum NLLR \tag{4.13}$$

Zero hypothesis

To account for the possibility that the target is not present in this scan, a *zero* hypothesis, or *dummy* hypothesis as it is sometimes called, is generated with the predicted state and covariance $\bar{\mathbf{x}}_1, \bar{\boldsymbol{P}}_1$. This node is numbered 0 in Figure 4.3, and $\bar{\mathbf{x}}_1$ in Figure 4.9.

Pure radar hypotheses

For every radar measurement (R1,R2 and R3 in Figure 4.6) inside the gate in (4.11), a new track hypothesis ($\hat{\mathbf{x}}_{R1}$ and $\hat{\mathbf{x}}_{R2}$ in Figure 4.6) is generated with filtered state and covariance according to the regular Kalman 'measurement update' equation (4.14).

$$\begin{aligned} \boldsymbol{K} &= \bar{\boldsymbol{P}}\boldsymbol{H}^T\boldsymbol{S}^{-1} \\ \hat{\mathbf{x}} &= \bar{\mathbf{x}} + \boldsymbol{K}\tilde{\mathbf{z}} \\ \hat{\boldsymbol{P}} &= (\boldsymbol{I} - \boldsymbol{K}\boldsymbol{H})\,\bar{\boldsymbol{P}} \end{aligned} \tag{4.14}$$

$$\begin{aligned} \boldsymbol{K} &= \text{Kalman gain} \\ \boldsymbol{S} &= \text{Covariance innovation} \\ \hat{\boldsymbol{P}}_1 &= \text{filtered state covariance} \end{aligned}$$

4.5 Process AIS measurements

As elaborated in Section 3.2 all AIS measurements are preprocessed to remove out-of-order messages and ID-swap errors, and only the latest AIS update from each target (MMSI number) are passed through to the MHT tracking loop. All AIS measurements outside the radar surveillance region are also removed from the measurement set.

The integration of AIS measurements into the MHT framework is not obvious and multiple approaches is possible. Since the AIS and radar measurement originate from different times, only sequential fusion methods [24] are considered in this work. The first step in any approach would be to decide which AIS measurements the leaf node shall consider. The radar measurements usually arrive at fixed intervals and is normally not synchronized to an external clock. AIS on the other hand is transmitted at asynchronous intervals and the messages are time stamped with UTC time in whole seconds. These properties leads to a finite and in most cases relatively small number of possible AIS time stamps in between each radar scan. The long runtime for each iteration for a MHT algorithm favours synchronous processing of both AIS and radar measurements at the arrival of each radar scan (synchronous processing). A choice that have to be made is if and when the AIS measurements should be gated. If no MMSI are associated in a track hypothesis the most natural way might be to gate the AIS measurements in a somewhat similar fashion as the radar measurements. However, if a track hypothesis have been associated with an MMSI previously, a natural approach could be to automatically associate this track hypothesis with new AIS measurement with the same MMSI. This could most

likely lead to a very good tracking as long as the MMSI to target association is correct, but would lead to a divergence between the following radar measurements for the actual target and the falsely associated AIS track. An alternative to the latest approach is to always gate the AIS measurements in some way and only allow track hypotheses which are already associated with an MMSI to only accept AIS measurements with that MMSI.

Since this MHT module is based on AIS aiding, the latest approach with continuous gating is chosen. This leads to two alternatives to gating; compare and gate AIS measurements at the AIS message times or predict the AIS measurements forward to the radar measurements time and gate at the same time as radar measurements.

Algorithm 1 AIS gating at AIS time

```
1:  procedure NODE::GATEAISMEASUREMENTS(AISmeasurements)
2:      M0 ← AISmeasurements
3:      aisTimes ← Set(M0.times)
4:      for time in aisTimes do
5:          T ← time − node.time
6:          x̄ = Φ(T)x_c
7:          P̄ = Φ(T)P_c Φ(T) + Q(T)
8:          M1 ← M0 where measurement.time == time
9:          accuracySet ← Set(M1.accuracy)    ▷ Accuracy can only be high or low
10:         for accuracy in accuracySet do
11:             M2 ← M1 where measurement.accuracy == accuracy
12:             S ← H P̄ H^T + R_accuracy
13:             for measurement in M2 do
14:                 z ← measurement.value
15:                 z̃ ← z − H x̄
16:                 NIS ← z̃^T S^{-1} z̃
17:                 aisInsideGate ← NIS ≤ η^2
18:                 if aisInsideGate then
19:                     Filter x̄ with measurement
20:                     Predict to radar time
21:                     for radarMeasurement in radarMeasurements do
22:                         Gate radar measurement as with pure radar measurements
23:                         for radarMeasurement inside gate do
24:                             Create fused node
```

Algorithm 1 outlines the main steps when gating at AIS times. The first step is to find all the different time stamps in the AIS measurement list. Since most maritime radars operate between 24 and 48 Rotations Per Minute (RPM) and the AIS time in seconds is always integer, a maximum of 2 different AIS times will exist in between two radar scans of a 24 RPM radar. Then for each time in the time set, a predicted state and covariance is calculated and the measurements that are stamped with this time is picked out of the AISmeasurement list. Since AIS messages transmit accuracy as either high or low, two different measurement covariances might be needed. The covariance residual is calculated for each accuracy among the AIS measurements, whereon the AIS measurements are gated, filtered and scored. The filtered state is then predicted to the radar time, where the predicted state and residual covariance are used to gate radar measurements as in Section 4.4. In the situation were no radar measurements is inside the gate, a pure AIS hypothesis is created at the predicted state (radar time). Whether or not it is desirable to create an AIS 'dummy' hypothesis / pure AIS hypothesis if there are any radar measurements inside the gate is a design choice. It will lead to a further increase in tree growth and computational cost, and might not lead to better tracking performance since the residual covariance after filtering with an AIS measurement would be relatively small and the difference between the pure AIS hypothesis and the fused hypotheses would be marginal.

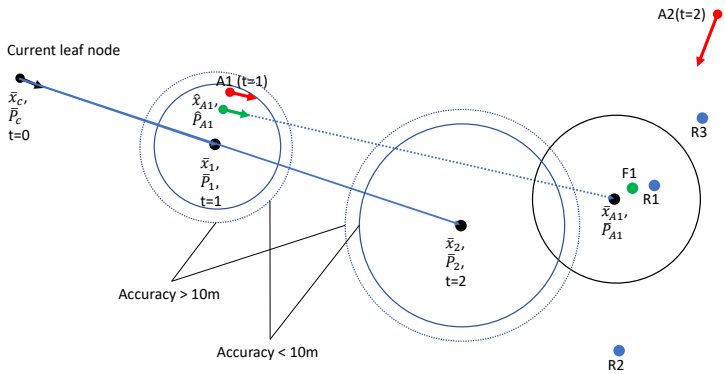

Figure 4.7: Gating AIS at AIS time

Figure 4.7 illustrates a situation where a single leaf node is showed in the upper left corner at $t = 0$, two red AIS measurements (A1 and A2) with position and velocity at $t = 1$ and $t = 2$ respectively and three blue radar measurements (R1, R2 and R3) at

$t = 2.5$. $\bar{\mathbf{x}}_\mathbf{1}$ and $\bar{\mathbf{x}}_\mathbf{2}$ are the predicted states for AIS messages at $t = 1$ and $t = 2$ respectively, and the solid blue circles are representing the high accuracy gates while the dotted blue circles are representing the low accuracy gates. In this scenario only A1 is inside the AIS gates, and a filtered state $\hat{\mathbf{x}}_{A1}$ and covariance $\hat{\boldsymbol{P}}_{A1}$ is calculated. $\hat{\mathbf{x}}_{A1}$ is then predicted to $t = 2.5$ ($\bar{\mathbf{x}}_{A1}$ and $\bar{\boldsymbol{P}}_{A1}$) and radar measurements are gated with the black solid circle based on the residual covariance to $\bar{\boldsymbol{P}}_{A1}$. Only R1 is inside this gate, thus a single fused hypothesis (F1) is created based on $\bar{\mathbf{x}}_{A1}$ filtered with R1.

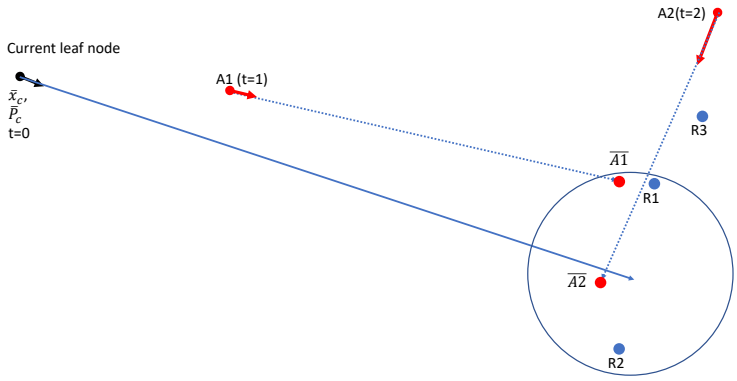

Figure 4.8: Gating AIS at radar time

If gating at radar time, all AIS measurements are predicted to radar time, and the predicted measurements are gated with the same gate at the radar measurements. By using the same gate on all measurements we assume that the radar measurement covariance is larger than the largest AIS measurement covariance, which is a reasonable assumption. This approach could however lead to unwanted AIS measurements inside the gate since the AIS measurements from other vessels can be predicted into the gate, thus leading to a more challenging association. This is exemplified in Figure 4.8, where AIS measurement A2 would be inside the gate when gating at radar time but not if gating at AIS time. For all AIS measurements inside the gate, a complete set of fused hypotheses are created. In our example this would be ($\overline{A1}$,R1), ($\overline{A1}$, R2), ($\overline{A2}$, R1) and ($\overline{A2}$,R2). Since pure radar hypotheses are created prior to AIS processing it is not necessary to create any new pure radar hypotheses.

In this work, the first approach is used for its assumed better performance and utilization of original data rather than predicted data. The AIS measurements are buffered between the radar scans, and pure AIS hypotheses are only created when no radar mea-

surements fall inside the gate.

4.5.1 Predict to AIS times

To gate the AIS measurements, we first have to predict the target state and covariance (4.16) for each of the AIS time stamps (4.15) in the received AIS measurements. Since AIS only transmits a single integrity status, better or worse than 10 meter, maximum two possible $\boldsymbol{R}_{AIS}$ must be considered. With maximum two different time stamps and two different accuracies, a maximum of four different gates must be considered.

$$\begin{aligned} \Delta T_{AIS_1} &= t_{AIS_1} - t_c \\ \Delta T_{AIS_2} &= t_{AIS_2} - t_c \end{aligned} \tag{4.15}$$

$$\begin{aligned} \bar{\mathbf{x}}_1 &= \boldsymbol{\Phi}(\Delta T_{AIS_1})\mathbf{x}_c \\ \bar{\boldsymbol{P}}_1 &= \boldsymbol{\Phi}(\Delta T_{AIS_1})\boldsymbol{P}_c\boldsymbol{\Phi}^T(\Delta T_{AIS_1}) + \boldsymbol{Q}(\Delta T_{AIS_1}) \\ \boldsymbol{S}_{1_{High}} &= \boldsymbol{H}\bar{\boldsymbol{P}}_1\boldsymbol{H}^T + \boldsymbol{R}_{AIS,High} \\ \boldsymbol{S}_{1_{Low}} &= \boldsymbol{H}\bar{\boldsymbol{P}}_1\boldsymbol{H}^T + \boldsymbol{R}_{AIS,Low} \\ \bar{\mathbf{x}}_2 &= \boldsymbol{\Phi}(\Delta T_{AIS_2})\mathbf{x}_c \\ \bar{\boldsymbol{P}}_2 &= \boldsymbol{\Phi}(\Delta T_{AIS_2})\boldsymbol{P}_c\boldsymbol{\Phi}^T(\Delta T_{AIS_2}) + \boldsymbol{Q}(\Delta T_{AIS_2}) \\ \boldsymbol{S}_{2_{High}} &= \boldsymbol{H}\bar{\boldsymbol{P}}_2\boldsymbol{H}^T + \boldsymbol{R}_{AIS,High} \\ \boldsymbol{S}_{2_{Low}} &= \boldsymbol{H}\bar{\boldsymbol{P}}_2\boldsymbol{H}^T + \boldsymbol{R}_{AIS,Low} \end{aligned} \tag{4.16}$$

4.5.2 Gate, filter and score

For each leaf node, all AIS measurements are gated with the gate matching their time, accuracy and threshold (Table 4.2). The measurements that pass the gating is then filtered with the predicted state and covariance matching its time, giving rise to an intermittent node (4.17).

$$\hat{\mathbf{x}}_1 = \bar{\mathbf{x}}_1 + \boldsymbol{K}\tilde{\mathbf{z}} \tag{4.17}$$

Confidence	70%	80%	90%	95%	97.5%	99%	99.5%
$\eta^2_{df=4}$	4.88	5.99	7.78	9.49	11.14	13.28	14.86

Table 4.2: Inverse χ^2 CDF for four degrees of freedom

Depending on how we want the AIS to affect our tracking, two different scoring strategies are explored. The first approach is to view the AIS as a pure *aiding*, where its only purpose is to improve the gating and uncertainty for the radar measurements. In this case, all AIS measurements are gated with a logarithmic score of zero, leading to neither improvement or worsening of the accumulative score of the track.

A second approach is to adapt the TOMHT score function [17] to the AIS paradigm. Since the AIS measurements are inherently labelled, then viewed from a single target, all AIS measurements except maximum one (depending on whether it have AIS transceiver or not) can be considered as clutter. If we then make the same assumptions as for radar measurements with respect to uniform spatial distribution and Poisson density distribution, we can estimate the expected number of AIS 'clutter' measurements λ_{AIS} based on the amount of targets with AIS transmitters and the observation area (4.18), where r_{radar} is the radar range. The clutter is in this context any measurement that does not belong to the target and which the target can erroneously utilize. The estimated clutter density could be calculated for a single frame, or averaged over a sliding window to reflect the AIS message flow over time since AIS transmission is not synchronised with the radar period. The resulting score function becomes (4.19).

$$\lambda_{AIS} = \frac{n_{AIS}}{\pi r_{radar}^2} \tag{4.18}$$

$$\mathrm{NLLR}_{AIS} = \frac{1}{2} NIS + \ln \lambda_{AIS} |2\pi S|^{1/2} \tag{4.19}$$

Testing has shown that both the first and last method gives an improvement over pure radar tracking, and since the last method gives a little more improvement since it scores the AIS measurements with a reasonable values compared to the radar measurements, hence not giving the AIS measurements an enormous advantage or disadvantage. The second method is used in all the simulations in Chapter 5.

4.5.3 Predict to radar time

All gated and filtered states from Section 4.5.2 is then predicted forward to the time of the radar measurements. This time delta will differ based on the time of the AIS measurement. Radar measurements are then gated for each predicted state and covariance, whereon a fused hypothesis are created for each radar measurement inside the gate. If there is no radar measurements inside the gate, a pure AIS hypothesis are created.

Fused hypotheses

The predicted state and covariance is then filtered with the gated radar measurement according to (4.14). The radar measurement is scored according to (4.12), and the hypothesis score is a weighted sum of the AIS and radar score (4.20).

$$\text{NLLR} = \frac{1}{2}\text{NLLR}_{AIS} + \frac{1}{2}\text{NLLR}_{Radar} \tag{4.20}$$

Pure AIS hypotheses

If no radar measurements are present in the gate, pure AIS hypotheses are created. This can be the situation when a target is broadcasting an AIS message, but is either in radar shadow or is not detected by the radar for any reason. These hypotheses are not created when one or more radar measurements are available, based on the assumption that if a radar measurement is present, the difference between a fused hypothesis and a pure AIS hypothesis is quite small since the AIS measurement covariance typically will be much smaller than the radar measurement covariance, leading to a fused state very close to the AIS measurement. The pure AIS hypothesis will use its predicted state and covariance and be scored with NLLR_{AIS}, somewhat similar to radar zero hypotheses.

4.6 Clustering

The problem of finding the globally optimal set of track hypotheses increases exponentially with the number of hypotheses in the problem. To reduce the size of the problem, it is desirable to split it into smaller independent problems. Both because it enables parallel computation and it reduces the total cost of solving the problem. Track trees that have common measurements must be solved together, since they can have mutual exclusive leaf nodes. The clustering can be done efficiently through Breath First Search (BFS) or Depth First Search (DFS) on a graph made from the track hypothesis tree.

By constructing a 0–1 adjacency matrix describing the connection between all the nodes in the track forest, the clustering problem is equivalent to the *connected components* problem in graph theory [33].

4.7 Optimal data association

The aim of this section is to elaborate the use of ILP to solve the data association problem in MHT that arises when there are multiple, possibly mutually exclusive, possibilities of measurement arrangements within the existing set of tracks. When the targets are divided into independent clusters, each of them can be treated as a global problem where we want to minimize the cost or maximize the score of the selected track hypotheses (leaf nodes). The selected track hypotheses must also fulfil the constraints, that each measurement can only be a part of one track, and that exactly one track hypothesis must be selected from each target. Since only binary values, selected or not selected, is possible for selection of hypotheses, the problem becomes an ILP. In the case where a cluster is only containing one target tree, the best hypothesis can be selected by running a search among the leaf nodes after the highest score, since none of the leaf nodes are excluding other leaf nodes in other target trees. This will often be the case for targets that are largely spaced out, and their gates are not and have not overlapped in a while. For any other case, where there are two or more targets in a cluster, the procedure in Section 4.7.1 must be carried out.

4.7.1 Integer Linear Programming

The essence of any optimization problem is a cost function and a set of constraints. In our problem, we want to select the combination of hypotheses (leaf nodes) that gives the highest score / lowest cost, while not selecting any measurement more than one time and ensure that we select minimum and maximum one hypothesis from each target.

Our score function is the sum of the selected node scores, and since we are using negative logarithmic scores the goal is to minimize the overall score, making the problem a minimum cost problem. Our cost vector $\mathbf{c}$ is made up from the scores of all the leaf nodes in the forest build by the trees clustered together, arranged in the order they are visited by a DFS. The accompanying selection vector $\boldsymbol{\tau}$ is of the same dimension with boolean values, where the selected hypotheses are value 1 and all other is value 0. These two together form the objective function (4.21).

$$\min_{\boldsymbol{\tau}} \quad \mathbf{c}^T \boldsymbol{\tau} \tag{4.21}$$

To ensure that we are selecting the same measurement maximum one time, a binary matrix $\boldsymbol{A_1}$ describing the association between nodes and measurements are created.$\boldsymbol{A_1}$

has as many rows are there are unique real measurement in the cluster forest and as as many columns as there are track hypotheses. All radar- and AIS measurement are real measurement, whereas dummy nodes do not contain any real measurements. Each rows in $\boldsymbol{A_1}$ represents a real measurements that exists in the cluster forest, and each column represents a track hypothesis. For each track hypothesis are the rows that represents the measurements used by that track hypothesis given value 1 (True) to indicate the usage. All other rows for that column are given value 0 (False) to indicate that it does not use these measurements. The order of the columns in $\boldsymbol{A_1}$ is the same as the rows in $\mathbf{c}$ and $\boldsymbol{\tau}$. Since we are only limiting a maximum of one usage of each measurement and no minimum, the constraint becomes an inequality constraint (4.22) where $\mathbf{1}$ is a vector of ones with the same dimension as $\boldsymbol{\tau}$.

$$\boldsymbol{A_1\tau} \leq \mathbf{1} \tag{4.22}$$

The second constraint we need to impose is that we need to select exactly one track hypothesis from each tree. This can be done by creating a boolean matrix $\boldsymbol{A_2}$ which describes the relationship between hypotheses and trees / targets. $\boldsymbol{A_2}$ will have as many rows as there are targets in the cluster, and columns as $\boldsymbol{A_1}$. The intersections between hypotheses and targets that belong together is value 1, all other is value 0. Since this constraint has a fixed requirement, it is formulated as an equality constraint (4.23) with $\mathbf{1}$ as in (4.22).

$$\boldsymbol{A_2\tau} = \mathbf{1} \tag{4.23}$$

The complete ILP formulation becomes (4.24), where $\boldsymbol{\tau}$ is a binary vector with dimension equal the number of leaf nodes in the track forest.

$$\begin{aligned} \max_{\boldsymbol{\tau}} \quad & \mathbf{c}^T\boldsymbol{\tau} \\ \text{s.t.} \quad & \boldsymbol{A_1\tau} \leq \mathbf{b_1} \\ & \boldsymbol{A_2\tau} = \mathbf{b_2} \\ & \boldsymbol{\tau} \in \{0,1\}^M \end{aligned} \tag{4.24}$$

An example based on Figure 4.9 at time step 2, where the $\boldsymbol{A}$ matrices and $\mathbf{C}$ vector would be (4.25).

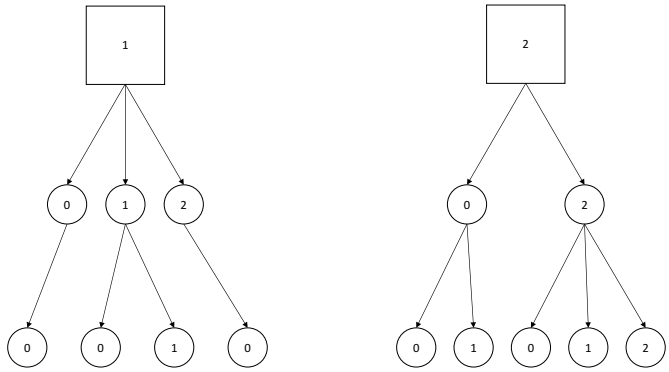

Figure 4.9: Track hypotheses forest

$$
\begin{aligned}
\boldsymbol{A_1} &= \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{b_1} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \\
\boldsymbol{A_2} &= \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}, \quad \mathbf{b_2} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\
\mathbf{c} &= \begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 & \lambda_4 & \lambda_5 & \lambda_6 & \lambda_7 & \lambda_8 & \lambda_9 \end{bmatrix}^T
\end{aligned}
\tag{4.25}
$$

4.7.2 Solvers

The problem (4.24) is formulated on standard form, which enables the use of existing off-the-shelf ILP solvers. There are a lot of off-the-shelf ILP and Mixed Integer Linear Programming (MILP) solvers on the marked, both free open source and commercial. The performance difference of some solvers were tested in [2], where the difference where found marginal, most likely because each optimization problem is relatively small and the initialization and preprocessing of the solver and problem played a significant part of the runtime compared to the actual solving. In this work, Google Optimization Tools is used as interface between the programming language and the solver. The default solver CBC was used exclusively in this work as its performance was on par with the others tested in [2].

4.8 N-Scan pruning

To keep the computational cost within reasonable limits, it is necessary to limit the amount of time steps backwards in time that the algorithm computes. This is done by removing all branches but the active track hypothesis at the current root node, and assign the one remaining node as new root node. This procedure is graphically explained in Figure 4.10, where a solid square frame indicates the current root node, and a dotted square frame indicates the new root node. The bold arrows in the figure represents the active track.

4.8.1 Dynamic window

For any MHT to be realistic over time it need to have a sliding window removing the unused hypotheses N steps back in time. The sliding window size (N) could be a static design parameter or a function of the runtime of that tree, which reflects the overall size of the tree. This enables the system to adapt its core parameters to guarantee its runtime demands. This scaling of N proved itself very efficient through testing and development, but is disabled in Chapter 5 for true comparison between different windows sizes.

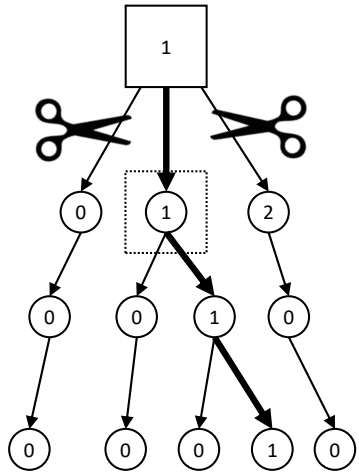

Figure 4.10: N-scan pruning

4.9 Track termination

Since targets can disappear from the observation region both by leaving the radar range and by driving behind objects that puts them in a radar shadow, it is necessary to terminate these tracks. It is also desirable to terminate falsely initiated tracks as soon as possible, since a guidance system would steer clear of any objects reported to it. The first scenario, where the target is leaving the radar range can easily be detected and the track can be terminated quickly based on the predicted position of the target relative to our own position. The second scenario, can be approached in different ways depending on the available data and computational power. The simplest solution is to terminate all tracks where the selected node after each iteration have a score higher that a threshold. This could terminate tracks with consecutive miss detections, which is desirable for false tracks, but can lead to premature termination of true targets with temporarily low P_D. This means that the termination threshold becomes a trade-of between killing false tracks and keeping targets with lower P_D and shadowed targets.

A more advanced approach could be to utilize map data to estimate whether or not a target is in a radar shadow of land objects, and then make a decision on whether this target should be given a lower P_D temporary or terminated based. This could also be done between targets if target extent is estimated, where targets behind other targets are given a lower P_D to punish miss detections less.

In this work, only range and score termination is implemented.

4.10 Track smoothing

After each iteration of the MHT algorithm a track list based on the selected hypotheses from Section 4.7 is passed forward to the operator and guidance system. Both human operators and guidance systems will try to predict the targets' behaviour based on their historical track. To improve the visualisation for this purpose it is possible to smooth the tracks based on the real measurements in the track and masking the dummy measurements in a Kalman smoother [34] with the same model as used in the predictions. As illustrated in Figure 4.12, this will lead to a smother and in most cases a more accurate representation of the true track since it avoids the straight lines caused by dead reckoning. The circles represent dummy measurements, while it is real measurements on both sides of the dead reconing period (not plotted to avoid cluttered illustration). Figure 4.11 is the same image zoomed in around the dead reconing area. This smoothing will however not affect the tracking performance since it is done after miss detections are corrected and only on the track list sent out of the tracking module.

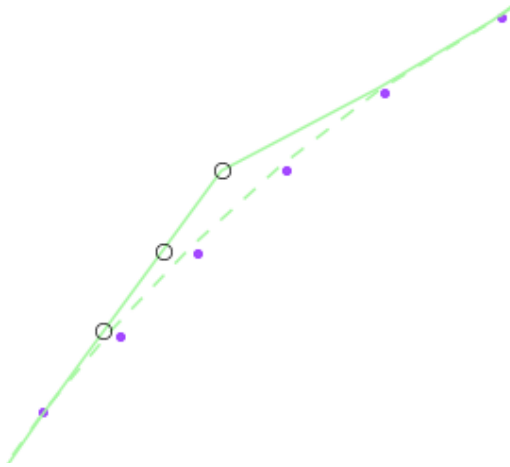

Figure 4.11: Track smoothing zoomed

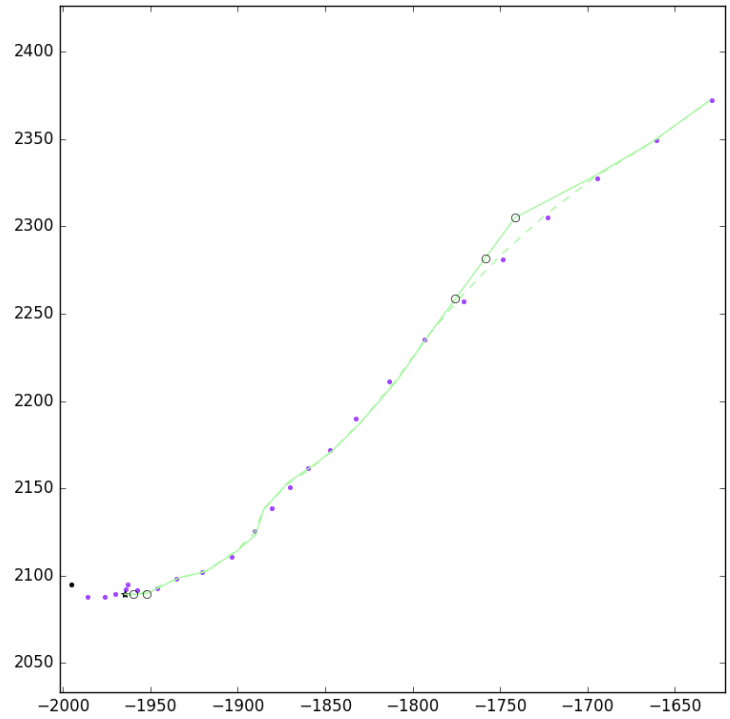

Figure 4.12: Track smoothing

Chapter 5

Results

The performance evaluation of any tracking system is difficult since the degrees of freedom are very large and there are no single obvious performance metric [35]. There are however two distinct testing methods found in the literature, pure Monte Carlo testing and situation/scenario testing. The first being that parameters like number of tracks, start position, velocity, manoeuvring, missed detections and clutter all are randomly selected and repeated many times. This approach is reasonable from a stochastic point of view, but it does not necessary create realistic tracking scenarios for in our case a maritime environment. The second approach is to create one or more scenarios which then is simulated with random variables like missed detections and clutter measurements. This approach is vulnerable to the created scenarios, since the design can heavily impact the measured performance. However, this method allows for construction of very specific situations where it is desirable to test multiple tracking system on the same custom created situation for comparison purposes.

The second approach is used in this work based on its ability to run user defined scenarios and its perceived more realistic behaviour.

5.1 Testing scheme

The performance evaluation of any MHT system is tedious in that it is necessary to test very many different situations to get a good understanding of how the system performs in a broad sense. The two largest factors contributing to the difficulty is the random nature of the clutter and lost detections. It is also desirable to evaluate the initialization

and tracking performance under both varying environmental (external) conditions and tuning (internal) setting. We want good tracking of targets with low probability of detection in cluttered environment, and have a runtime no larger than the radar rotation period. The initialization module must be able to detect targets with probability of detection lower than unity without initializing too many false tracks into the MHT algorithm. The testing following is separated into two parts; initialization and tracking.

5.2 Scenario

All simulations in this work is based on a generated scenario, shown in Figure 5.1, with black dots marking the initial time and position of the targets. The radar range is 5500 meter (~3 Nautical Miles (NMs)), which gives an area of surveillance of approximately 95 square km. The scenario contains 16 targets, which all starts inside the observable area of the radar. The scenario contains a mixture of fast and slow moving vessels, some with sharp turns and some almost at stand still. Table 5.1 show the initial states of all targets, and the true path is generated once from these initial values.

From this base scenario, five scenarios where generated with different AIS configuration on the vessels, see Table 5.2. The first scenario represents the baseline with only radar information available, whereas the rest have some level of AIS information. Scenario 1 adds 50% class B AIS transmitters, and is representing a situation where all the targets are smaller vessels with some voluntarily installed AIS transceivers. In scenario 3, all vessels have AIS class B installed. This scenario represents a best case situation regarding yacht and leisure vessels from an autonomous anti collision perspective and is only realistic if AIS class B were to be mandatory for these vessel classes. Scenario 2 is the same as scenario 1, with the difference that the vessels have class A transmitters in stead of class B. This gives them higher and smarter rate of transmission (see Tables 2.4 and 2.5), which in theory should improve tracking. This scenario can be viewed as a few commercial vessels travelling in between a large group of yachts. The last scenario, where all targets are equipped with class A transmitters is the ultimate situation for any fusion tracking system. This case would be realistic in a crowded professional working area, for instance harbours, fishing areas and off-shore installations.

Figure 5.2 show all radar measurements for an entire scenario for the different clutter level overlaid.

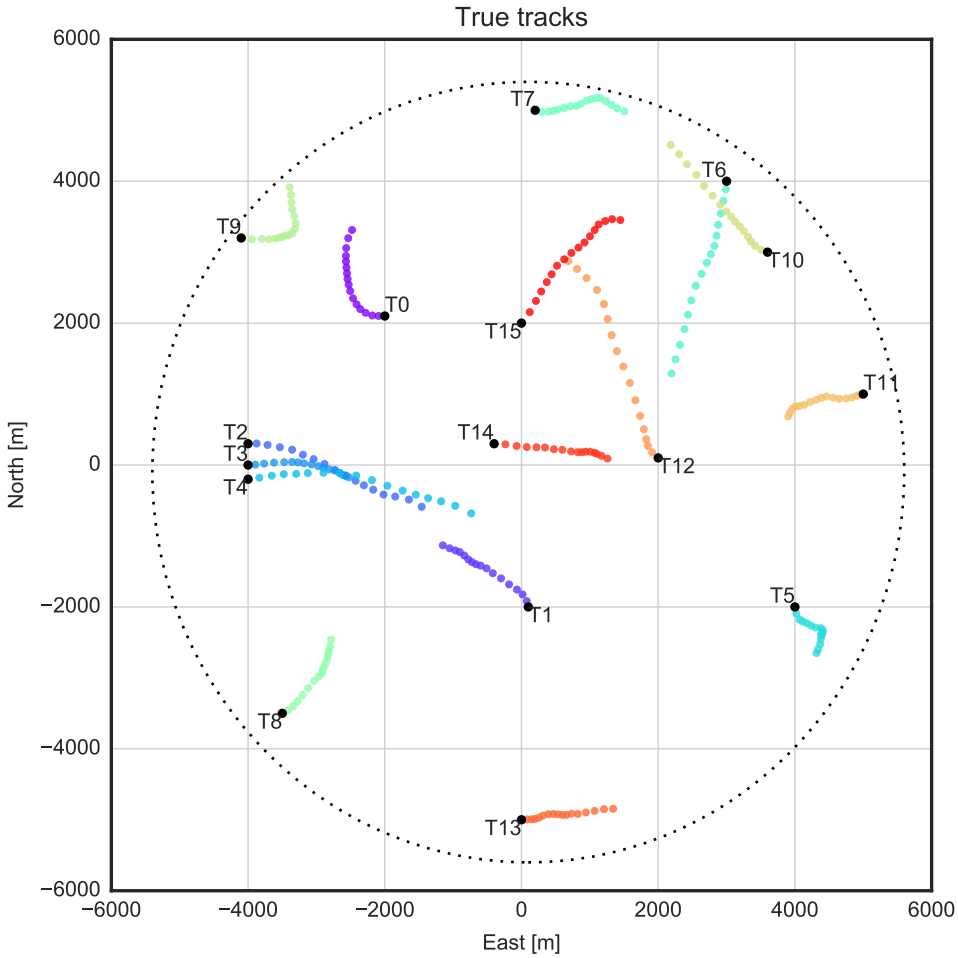

Figure 5.1: True tracks

Target	North	East	North speed	East speed
0	2100.0	-2000.0	10.0	270.0
1	-2000.0	100.0	8.0	-2.0
2	300.0	-4000.0	-1.0	12.0
3	0.0	-4000.0	12.0	90.0
4	-200.0	-4000.0	1.0	17.0
5	-2000.0	4000.0	-8.0	1.0
6	4000.0	3000.0	-8.0	2.0
7	5000.0	200.0	-1.0	10.0
8	-3500.0	-3500.0	5.0	10.0
9	3200.0	-4100.0	-2.0	17.0
10	3000.0	3600.0	3.0	-10.0
11	1000.0	5000.0	-2.0	-7.0
12	100.0	2000.0	8.0	-10.0
13	-5000.0	0.0	2.0	10.0
14	300.0	-400.0	0.0	17.0
15	2000.0	0.0	15.0	15.0

Table 5.1: Initial states

T	S0	S1	S2	S3	S4
0	-	B	A	B	A
1	-	-	-	B	A
2	-	B	A	B	A
3	-	-	-	B	A
4	-	B	A	B	A
5	-	-	-	B	A
6	-	B	A	B	A
7	-	-	-	B	A
8	-	B	A	B	A
9	-	-	-	B	A
10	-	B	A	B	A
11	-	-	-	B	A
12	-	B	A	B	A
13	-	-	-	B	A
14	-	B	A	B	A
15	-	-	-	B	A

Table 5.2: AIS class scenario configuration

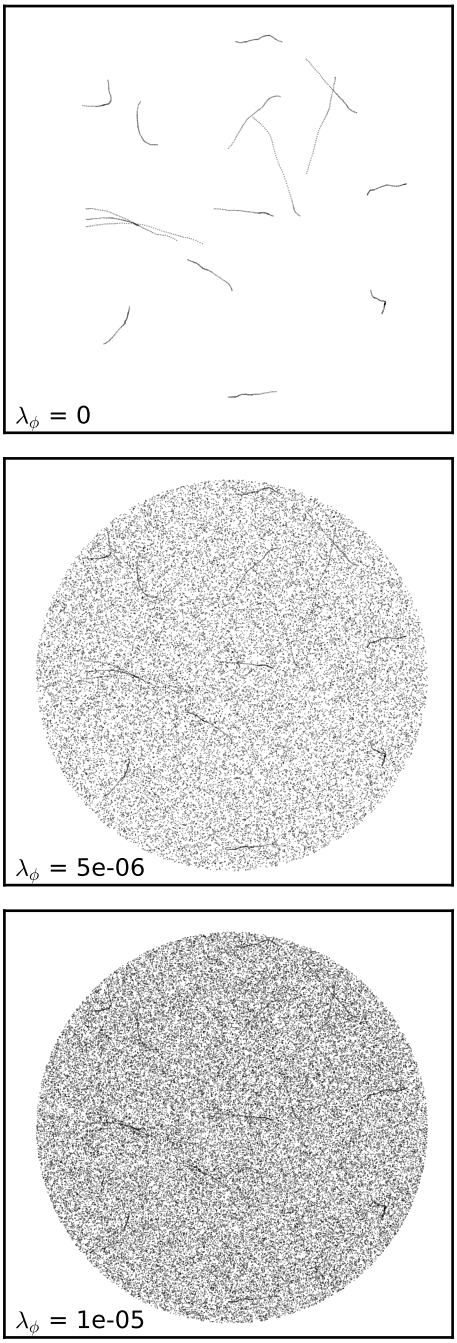

Figure 5.2: Scenario measurements overlaid

5.3 Simulation

Both initialization- and tracking performance are averaged over a set of 25 Monte Carlo simulations with differently seeded clutter- and missed radar detections. All simulations were done with a sampling interval at 2.5 seconds (24 RPM), which is a common rotation speed on a coastal maritime radar. Each of the 540 initialization variations and 180 track performance simulations where run on a dual Intel i7−6700 server running Linux Ubuntu with Solid State Storage (SSD) storage and 64 GB Random Access Memory (RAM).

5.4 Initialization results

The first performance metric for the initialization module is how long time it takes to initialize the correct tracks, which is tested under a range of internal and external conditions, see (5.1). All 108 combinations of these parameters were simulated on all five scenarios (Table 5.2), which are the same routes but with different AIS configurations. Each of these 540 variations were repeated 30 times, and from these 16,200 simulations, the average time to initiate true targets and amount of erroneous targets are calculated.

$$\begin{aligned} \mathbf{P_D} &= \begin{bmatrix} 0.9 & 0.8 & 0.6 \end{bmatrix} \\ (m/n) &= \begin{bmatrix} (1/1) & (1/2) & (1/3) & (1/4) \\ (2/2) & (2/3) & (2/4) & (2/5) \\ (3/3) & (3/4) & (3/5) & (3/6) \end{bmatrix} \\ \boldsymbol{\lambda_\phi} &= \begin{bmatrix} 0 & 5 \cdot 10^{-6} & 1 \cdot 10^{-5} \end{bmatrix} \end{aligned} \tag{5.1}$$

A track is categorized as correctly initialized if the position difference between the true track and the initial track (5.4) is less than a threshold. All initial tracks that do not correspond to true tracks, are categorized as erroneous. To analyse the impact of the erroneous tracks, the lifespan of falsely initiated tracks are investigated to see whether they die out at the same rate as they are initiated, or if they accumulate.

Figures B.1 to B.12, figs. B.13 to B.24, figs. B.25 to B.36, figs. B.37 to B.48 and figs. B.49 to B.60 show the initialization performance for Scenario 0 to 4 respectively, averaged over the Monte Carlo simulations. Their first plot is the share of true track correctly initialized over time, whereas the second and third plot is the amount and density of falsely initiated tracks. The share of correctly initialized tracks is defined as the number of true tracks that have been initialized minimum one time divided on the total number of true tracks. This

prevents re-initialized true tracks to be counted more than one time, and an initialization share on 100% means that all tracks have been initialized correctly at least one time. The difference between the two last plots is that the first is an accumulation of false track over time, while the last is the number of erroneous tracks alive at that time. These data are summarized in Figures 5.3 to 5.7 to better evaluate their performance over the set of variations.

For each scenario and initialization setting (m/n), an average cost of the variations (P_D and λ_ϕ) were calculated based on (5.2). $t_{80\%}$ is the time it took to reach 80% correctly initialized true tracks, and variations that never reaches 80% is penalized with a high 'initialization time' selected to match the worst scenarios. A lower cost means that the time it took to correctly initialize 80% of the true tracks and the average number of false tracks alive where both low. An increase in either time or number of false tracks would increase the cost. To better see the nuances close to zero, the plot show the normalized square root of the cost, were the normalization constant is the highest cost among all the scenario's tuning parameters. The normalization is done to keep in numeric values in a defined set since the cost function is only a representation of the performance between the different tuning setting.

$$Cost = t_{80\%}(1 + n_{FalseTrack}) \tag{5.2}$$

From Figures 5.3 to 5.7 we see that each M row has a peak (low cost) and edges with higher costs. (2/3) and (2/4) are overall good values with a balance between initialization time and the amount of erroneous tracks. We can also see that the AIS aided initializations are all performing better than the pure radar scenario.

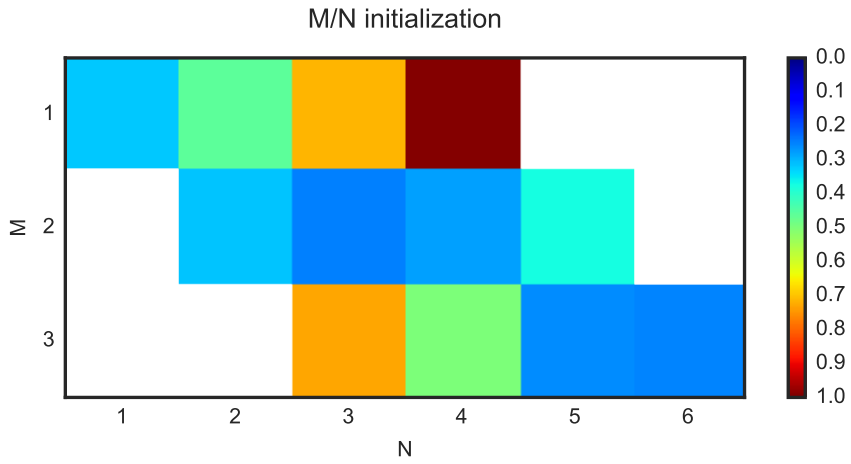

Figure 5.3: Scenario 0 — Initiator performance

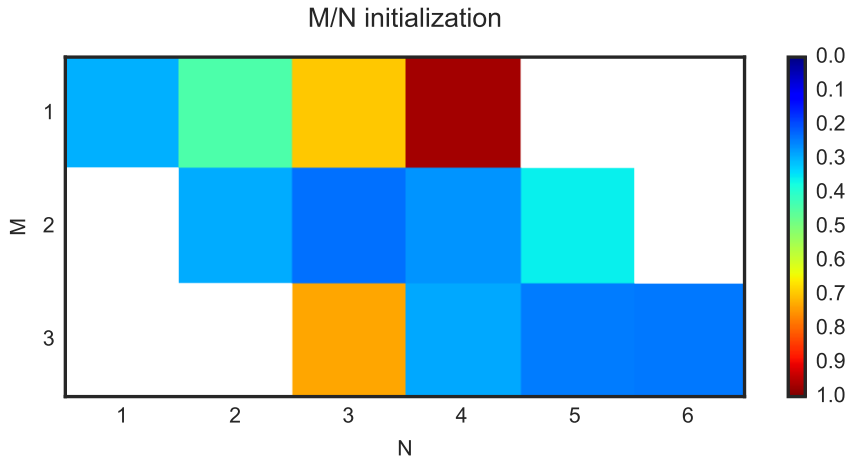

Figure 5.4: Scenario 1 — Initiator performance

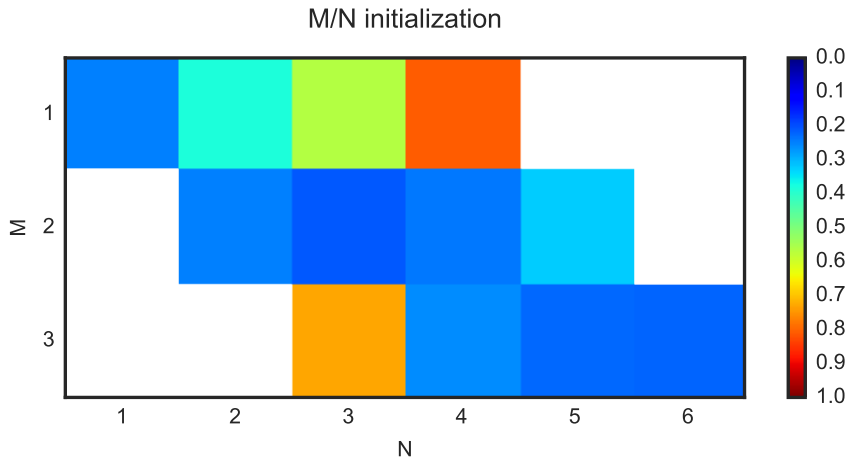

Figure 5.5: Scenario 2 — Initiator performance

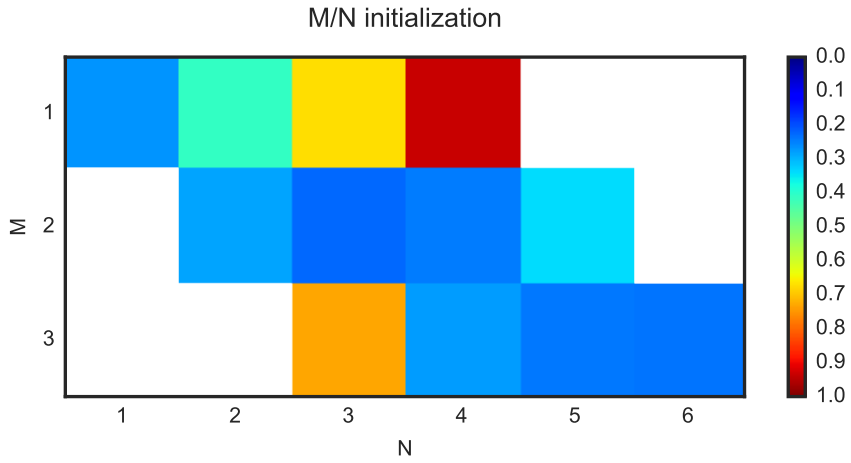

Figure 5.6: Scenario 3 — Initiator performance

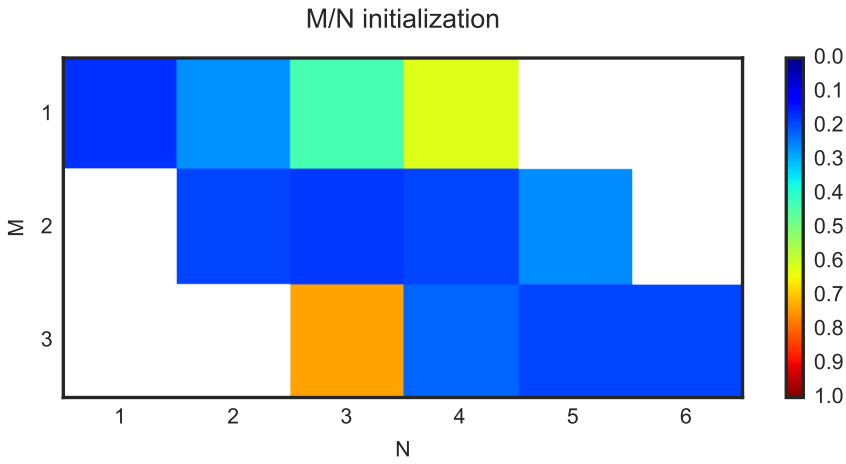

Figure 5.7: Scenario 4 — Initiator performance

5.5 Tracking results

When testing the tracking performance, it is desirable to remove the variable of initialization to better see difference in *tracking* rather than *initialization*. All simulations testing tracking performance are carried out with all targets correctly initialized at initial time, and with the initiator set to $(2/4)$. The unused measurements from the tracking algorithm is processed by the initialization algorithm, giving lost targets a chance to get re-initialized, which is an important property for any safety critical system. For scenarios where AIS measurements are available, the unused AIS measurements are used to aid the initiator. Similar to the initialization testing, each of the five scenarios are tested under varying internal and external conditions (5.3). The 180 variations are repeated 30 times, giving rise to 5400 simulations from which the performance metrics are calculated.

$$\begin{aligned} \mathbf{P_D} &= \begin{bmatrix} 0.9 & 0.8 & 0.6 \end{bmatrix} \\ \mathbf{N} &= \begin{bmatrix} 1 & 3 & 6 & 9 \end{bmatrix} \\ \boldsymbol{\lambda_\phi} &= \begin{bmatrix} 0 & 5 \cdot 10^{-6} & 1 \cdot 10^{-5} \end{bmatrix} \end{aligned} \tag{5.3}$$

Since the targets are initialized perfectly in every situation, we are interested in how good our system is able to *maintain* the tracks. We measure this by means of the Euclidean distance between the estimated and true track (5.4). The first track performance metric is the track loss percentage, where a track is considered correct as long as $\Delta P \leq \varepsilon_p$ for a given threshold. If a track is deviating more than the threshold and never returns within the threshold again, it is considered lost at the time-step when it exceeded the threshold. If the track should converge after exceeding the threshold, it is considered restored at the time-step it is returning within the limit. Tracks that deviates more than $10\varepsilon_p$ are considered lost at the time they exceeded ε_p. This two step threshold is to allow tracks to dead reckon for a while without being registered as lost, while still dismiss tracks that are deviating away from the true track.

$$\Delta P = \|\mathbf{p}_{track} - \mathbf{p}_{target}\|_2 \tag{5.4}$$

The second and closely related metric is the tracking percentage, where the total time a target is correctly tracked is summed up and compared with the existence time of the target. This situation is illustrated in Figure 5.8 where a track is lost at (-2500,2600), and re-initialized at (-2500,2650). The track is considered lost at (-2500,2600), while the

tracking percentage is also accounting the last track from (-2500,2650). This metric gives a more realistic measure of how likely targets are to be tracked, since lost tracks are assumed to often be re-initialized.

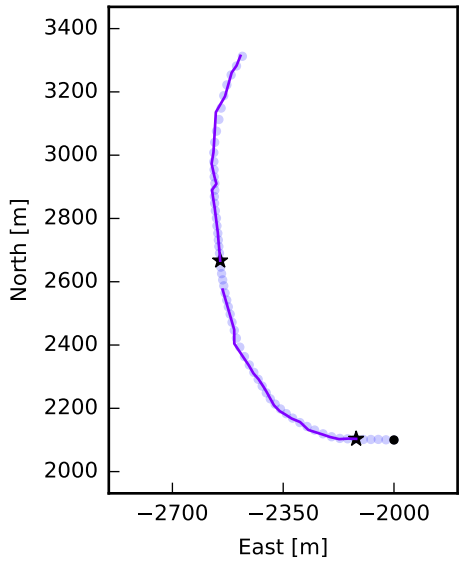

Figure 5.8: Tracking percentage, the sum of the total tracking time relative the existence time of the target

The third metric is how close the estimated track is to the true track on average. This is measured as the Root Mean Square Deviation (RMSD) of the entire track (5.5), where n is the length of the track.

$$RMSD = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(\Delta P_t)^2} \tag{5.5}$$

5.5.1 Track loss

From figs. 5.9 to 5.13 we can see that the amount of lost tracks are very dependent on the probability of detection P_D and window size (N). An interesting observation is the declination of lost tracks with increased clutter when N=1, which is most likely caused by clutter being gated and selected, giving the track a high enough score to not be terminated while still being similar enough to not loose the true track.

Table 5.3 show the improvement in track loss of the different AIS scenarios compared to the pure radar scenario, all for $\lambda_\phi = 1 \cdot 10^{-5}$. From this we can see that that the class of AIS has a large effect on the improvement when comparing the gain of all targets being equipped with class A versus class B AIS. It can also be seen that a minimum window size is necessary to achieve a substantial gain, this is probably because the tracking percentage is increasing but because of the low P_D and the varying nature of AIS report interval most of the targets will still be lost at least one time.

Pd	N	S1	S2	S3	S4
50%	1	0%	1%	0%	2%
50%	3	5%	46%	12%	94%
50%	6	26%	52%	50%	90%
50%	9	26%	50%	58%	85%
70%	1	1%	13%	1%	28%
70%	3	13%	49%	26%	93%
70%	6	12%	45%	27%	86%
70%	9	10%	45%	20%	84%
90%	1	4%	36%	9%	61%
90%	3	17%	50%	33%	50%
90%	6	18%	45%	36%	45%
90%	9	18%	45%	36%	45%

Table 5.3: Track loss improvement relative pure radar

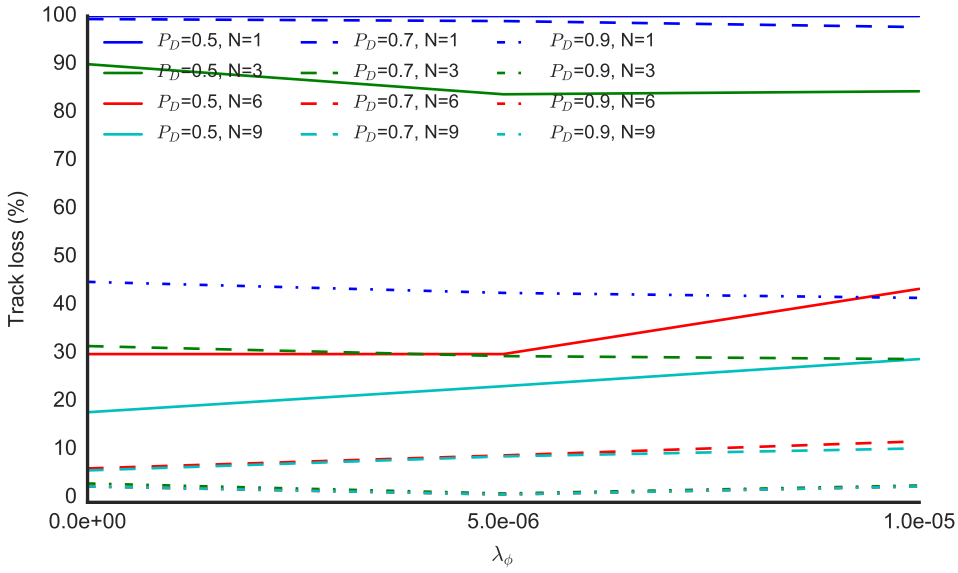

Figure 5.9: Scenario 0 — Track loss

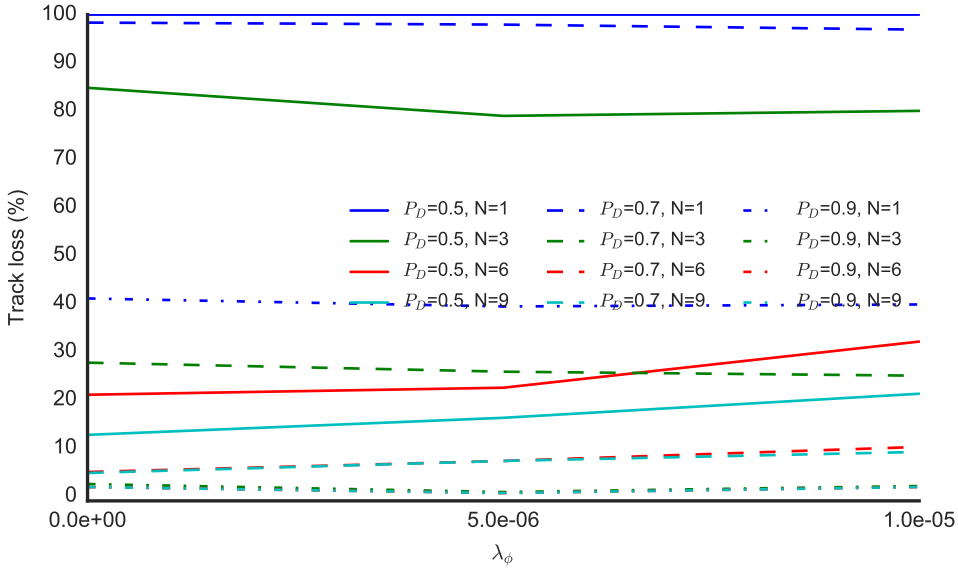

Figure 5.10: Scenario 1 — Track loss

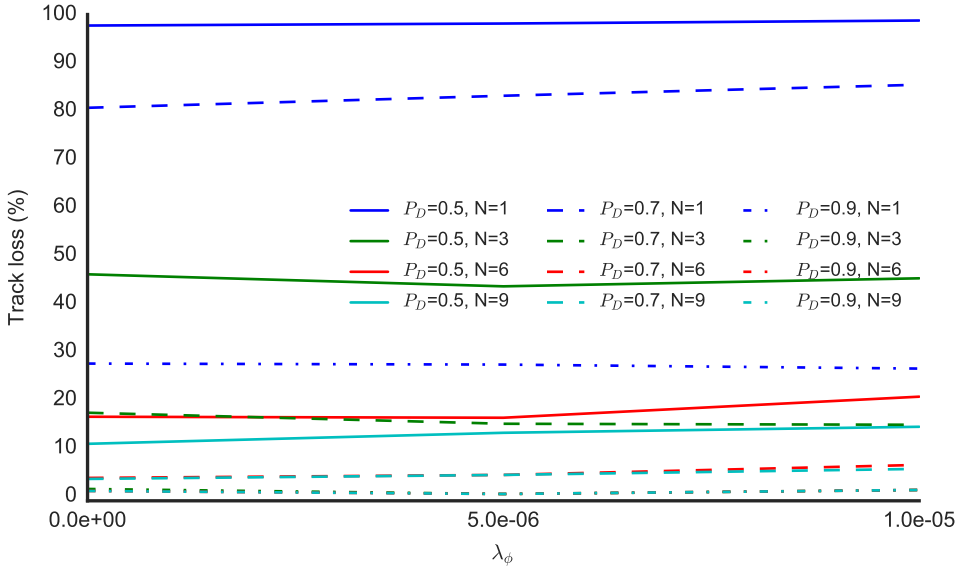

Figure 5.11: Scenario 2 — Track loss

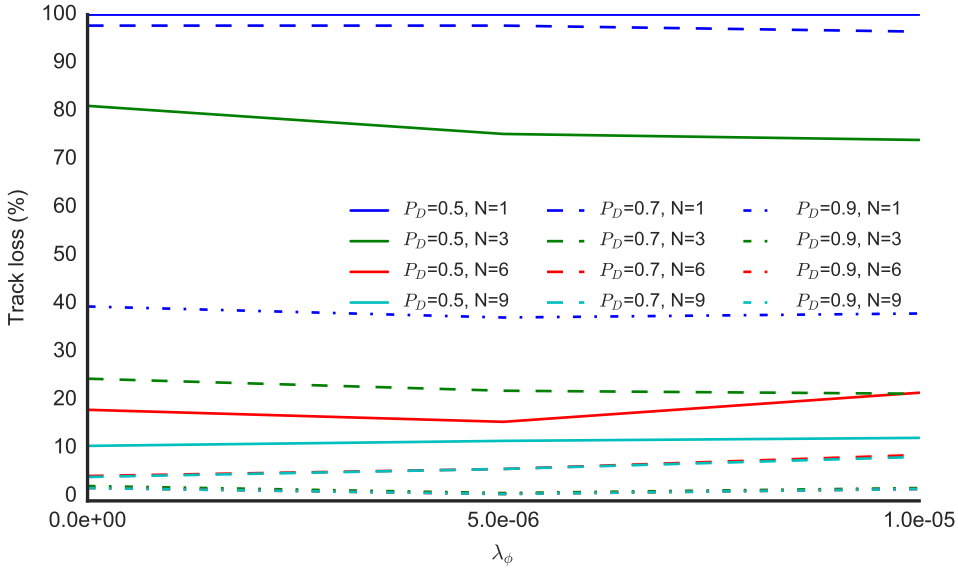

Figure 5.12: Scenario 3 — Track loss

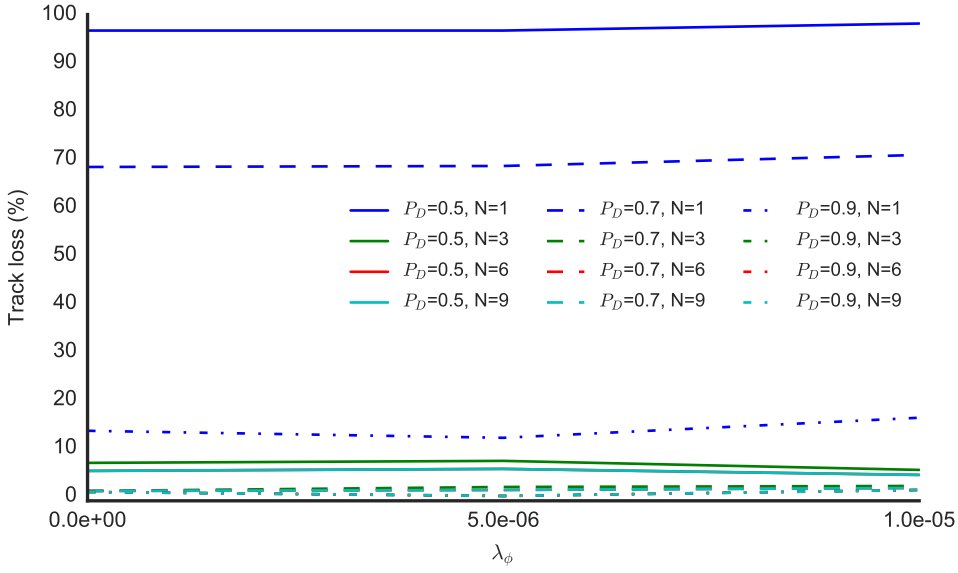

Figure 5.13: Scenario 4 — Track loss

5.5.2 Tracking percentage

Figures 5.14 to 5.18 show the tracking percentage for scenario $0 - 4$. From these we can see a substantial increase in the tracking percentage for low P_D and small window size when comparing the pure radar tracking with the AIS aided tracking.

Table 5.4 list the improvements for the AIS scenarios compared to the pure radar baseline, all for $\lambda_\phi = 1 \cdot 10^{-5}$. From this we can see that the class of AIS equipment impacts the tracking performance more than the amount of AIS transmitters. It also shows that the benefits of AIS decreases with an increased P_D and N. The performance gain for 70% P_D and higher is marginal for all cases except N=1 with class A AIS.

Pd	N	S1	S2	S3	S4
50%	1	17%	69%	29%	129%
50%	3	11%	24%	17%	43%
50%	6	6%	9%	10%	15%
50%	9	3%	5%	6%	8%
70%	1	8%	20%	11%	37%
70%	3	1%	2%	2%	4%
70%	6	0%	1%	0%	1%
70%	9	0%	0%	0%	1%
90%	1	1%	2%	1%	4%
90%	3	0%	0%	0%	0%
90%	6	0%	0%	0%	0%
90%	9	0%	0%	0%	0%

Table 5.4: Tracking percentage improvement relative pure radar

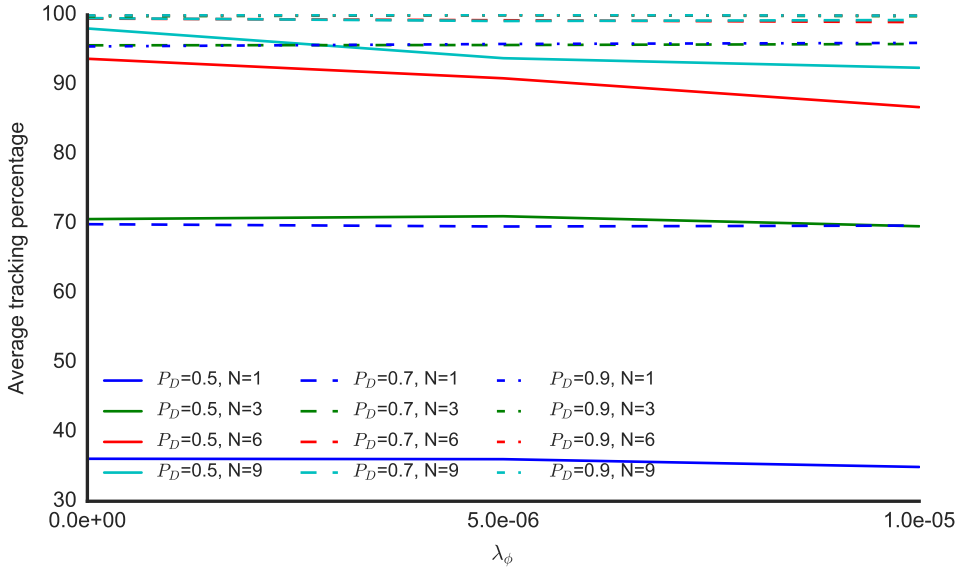

Figure 5.14: Scenario 0 — Tracking percentage

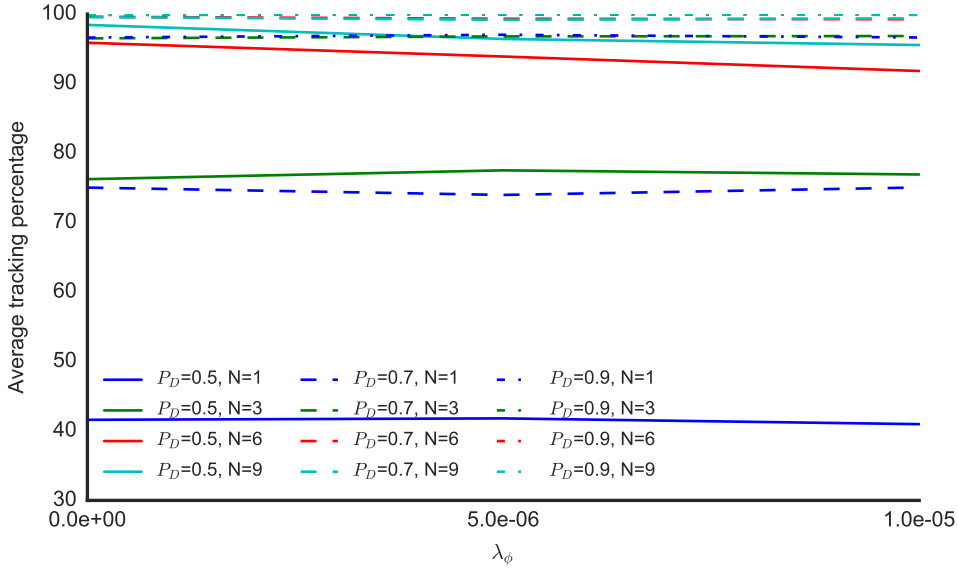

Figure 5.15: Scenario 1 — Tracking percentage

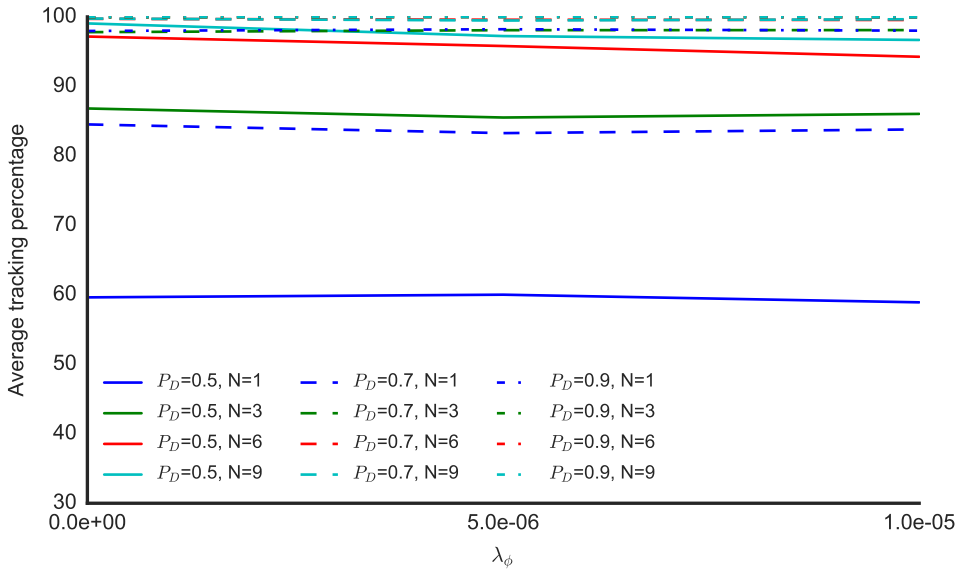

Figure 5.16: Scenario 2 — Tracking percentage

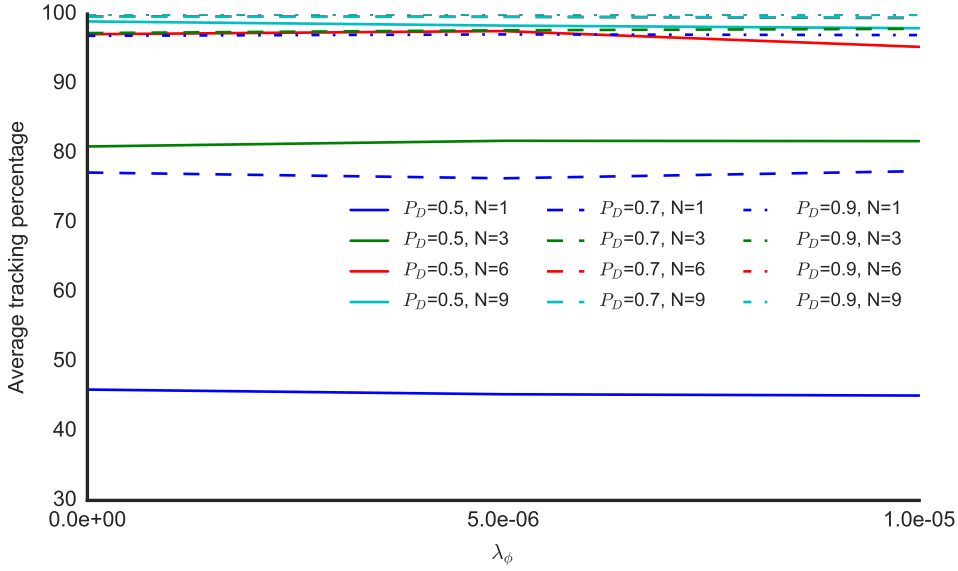

Figure 5.17: Scenario 3 — Tracking percentage

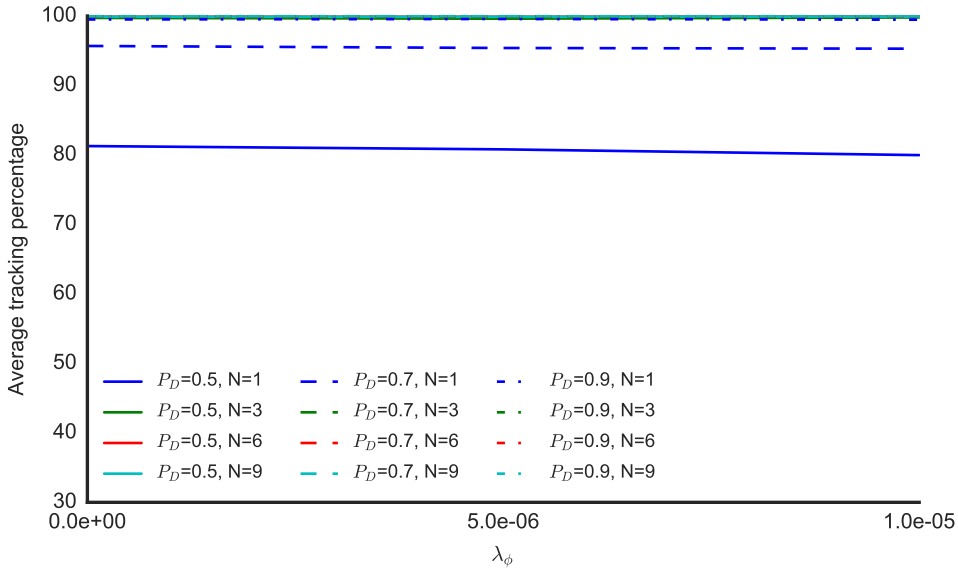

Figure 5.18: Scenario 4 — Tracking percentage

5.5.3 Effect of smoothing

Figures 5.19 to 5.23 show the Root Mean Square (RMS) error between the true and estimated tracks for $\lambda_\phi = 1 \cdot 10^{-5}$ with and without Kalman smoothing. From these we can see that the larger window sizes have higher errors, most likely since they are dead reckoning for a longer period and still able to converge back to the true track. We can also see that a larger window size gains more on smoothing and leads to better smoothed track overall, giving the CAS more accurate tracks to work on.

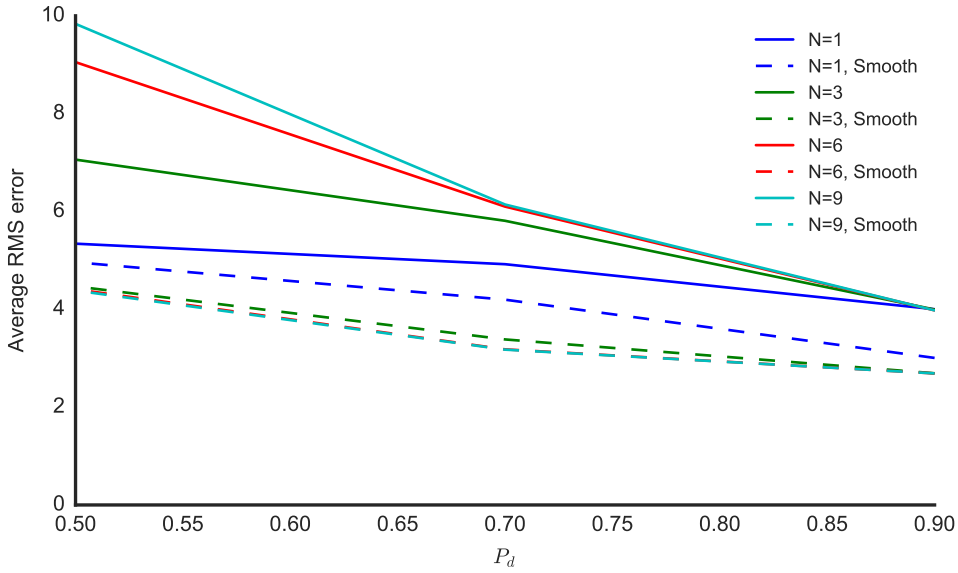

Figure 5.19: Scenario 0 — Tracking Correctness

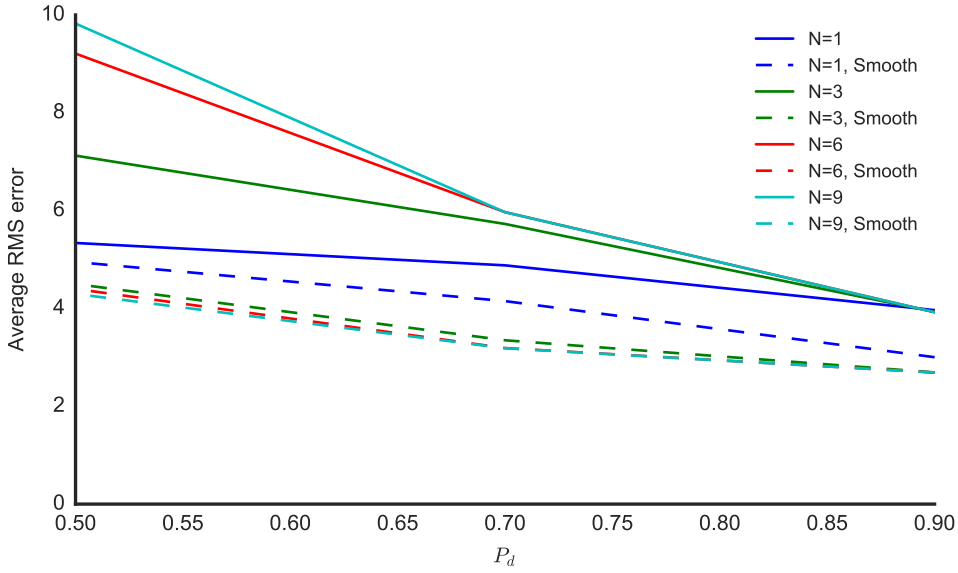

Figure 5.20: Scenario 1 — Tracking Correctness

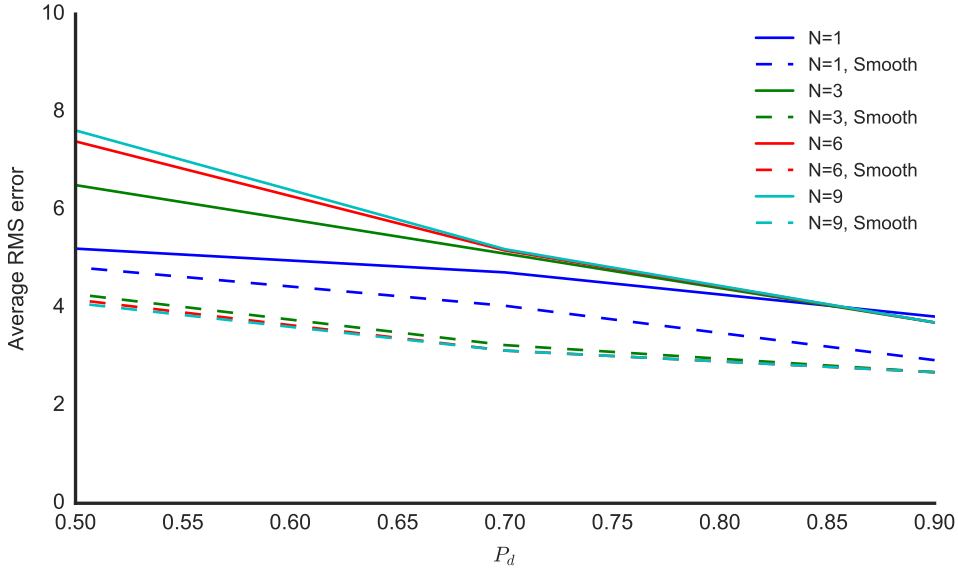

Figure 5.21: Scenario 2 — Tracking Correctness

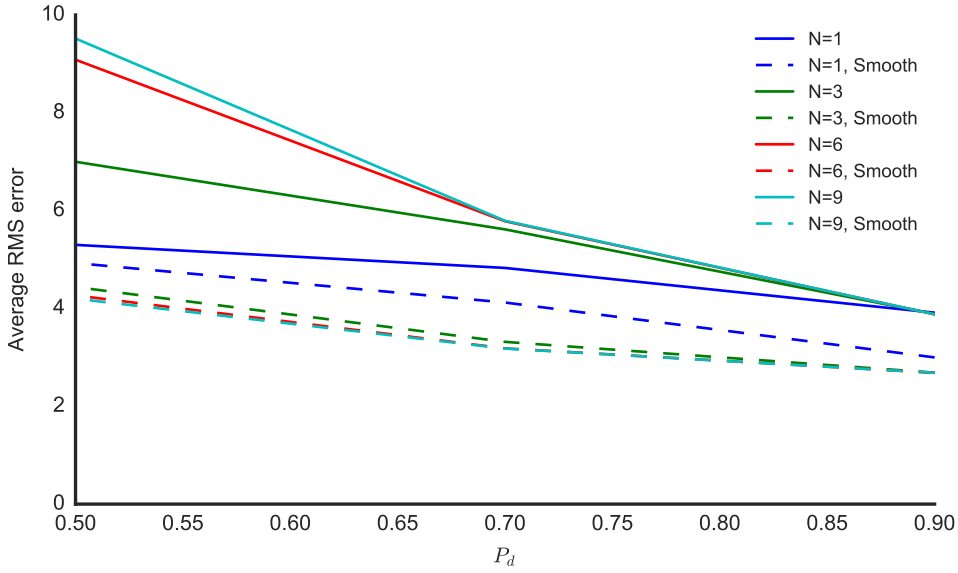

Figure 5.22: Scenario 3 — Tracking Correctness

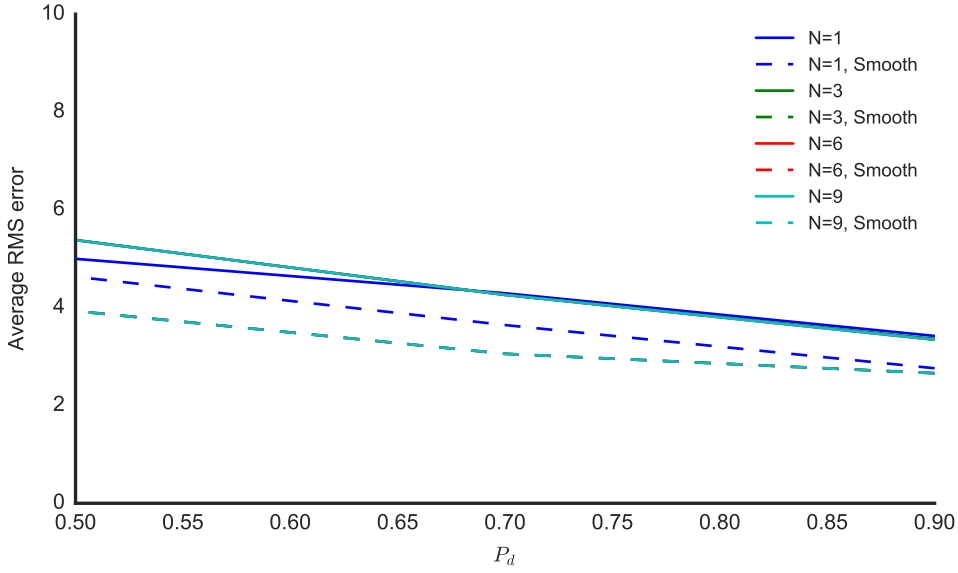

Figure 5.23: Scenario 4 — Tracking Correctness

5.5.4 Runtime performance

Figures 5.24 to 5.28 show the average runtime for an iteration for scenario 0 — 4. From these we can see the very expected increase in runtime as the window size increases, and that the AIS scenarios have a steeper increase that the pure radar scenario. This is caused by an increase in the number of nodes in the forest and a not as fast implementation of the AIS measurements processing as radar measurements, primarily caused by the large amount of steps in the AIS processing.

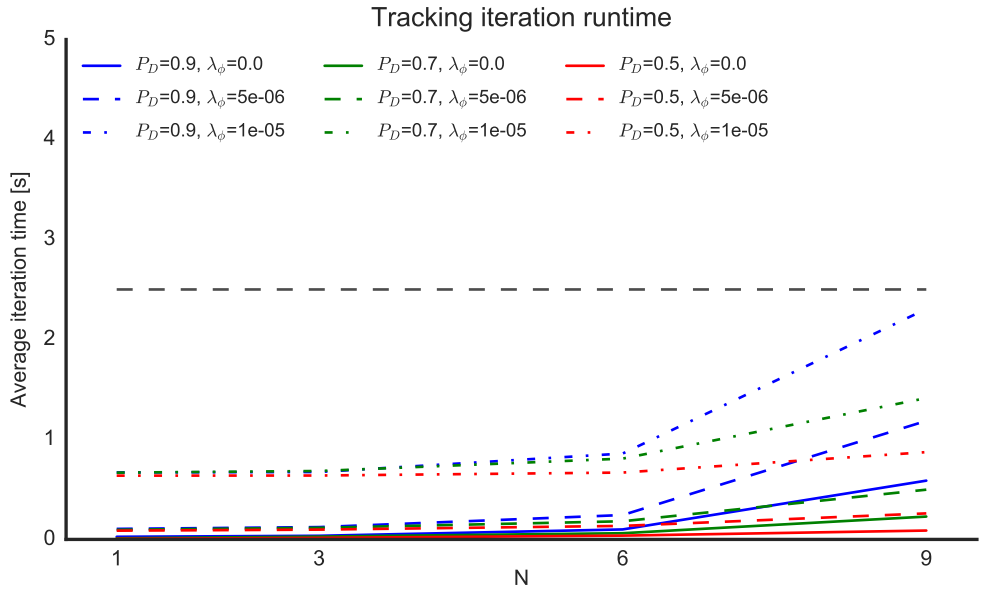

Figure 5.24: Scenario 0 — Tracking runtime

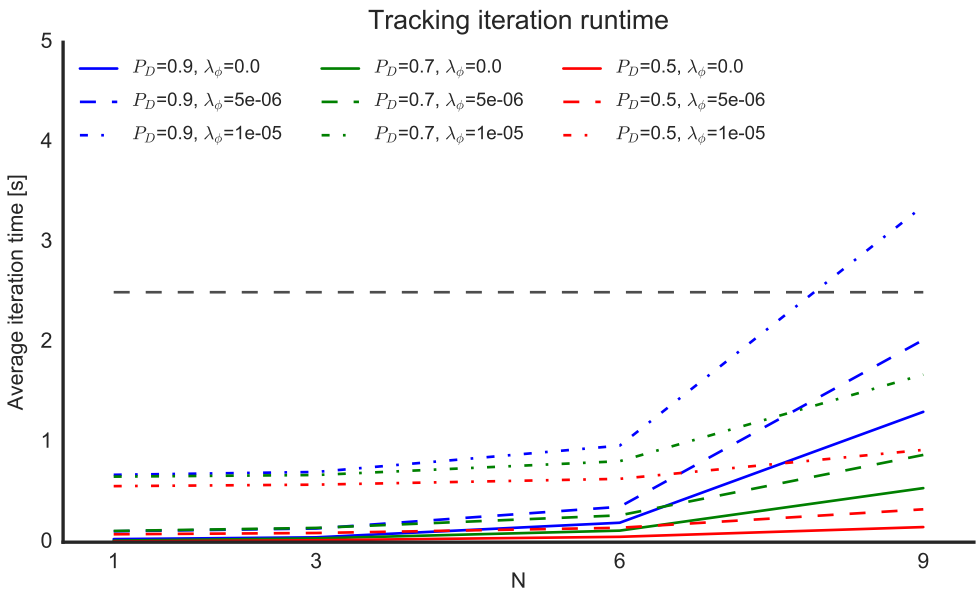

Figure 5.25: Scenario 1 — Tracking runtime

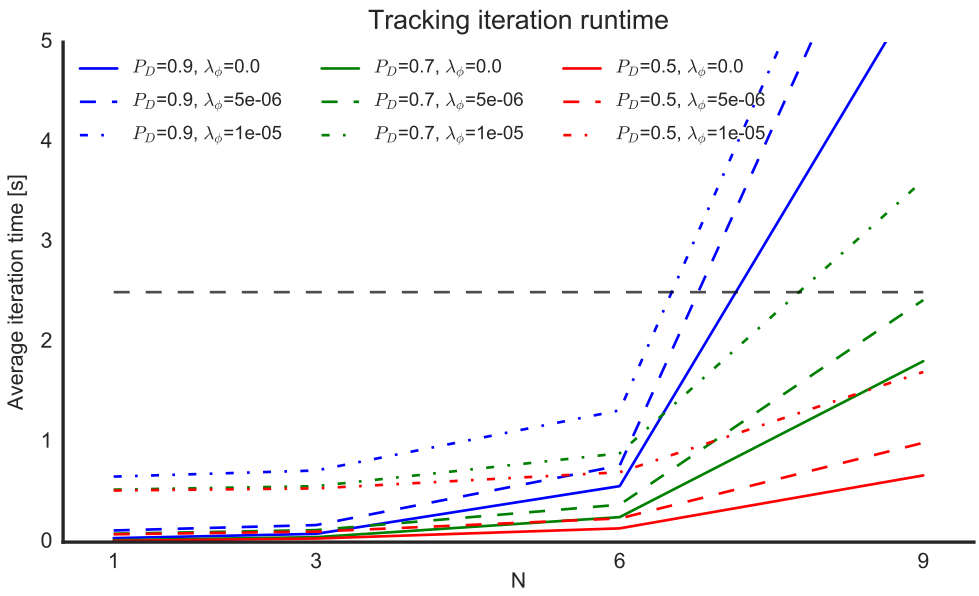

Figure 5.26: Scenario 2 — Tracking runtime

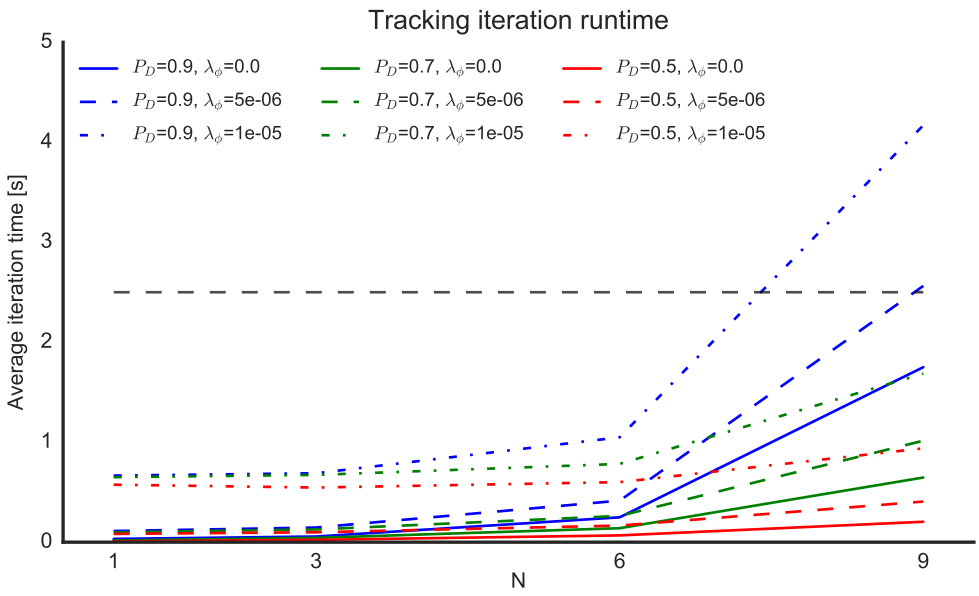

Figure 5.27: Scenario 3 — Tracking runtime

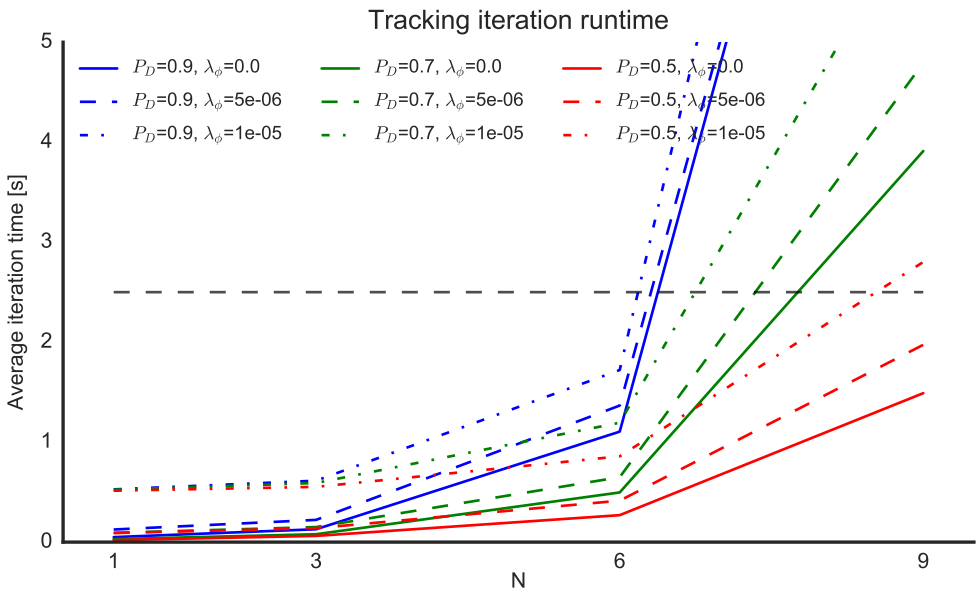

Figure 5.28: Scenario 4 — Tracking runtime

Chapter 6

Discussion

The 2/2&m/n initialization logic have proved to be a simple and reliable way to incorporate initialization to TOMHT. Its performance is tuning dependent and the simulation results indicates that there are multiple alternatives which all gives satisfactory results, though with a trade off between initialization time and amount of erroneous tracks.

The marginal increase in tracking percentage of medium to high P_D and N variations when aiding the tracking system with AIS can indicate that the AIS aiding will have greatest impact in situations where a vessels is in a temporary radar shadow. This is situations where a target for a limited time have a reduced P_D caused by blocking objects other vessels and land or masking phenomena like heavy snow or rain. Kalman smoothing of the outgoing track list have showed to provide a better estimate of the true tracks for all window sizes, and with a greater gain for larger window sizes.

6.1 Future work

There are several unanswered questions that need to be addressed in order to achieve a full utilization of the AIS information. The maybe most important is what to do when a track previously associated with an MMSI no longer has new AIS measurements from that MMSI inside its gate? This is a natural situation since the radar update period is often smaller that the AIS update period, but for how long time shall that track continue to be associated with that MMSI. And equally important is the decisions that needs to me made when the associated MMSI is appearing outside the gates for that track. One option is to always make hypothesis where the associated MMSI measurements is, but

this could lead to large jumps in the target state and if the associated MMSI was wrong, this would force a target of its radar track and lock it onto an AIS track.

An other open question is how to utilize the AIS meta-data to improve the tracking? From AIS length information, it might be possible to estimate the maximum turning rate for a vessel, which can be used to more accurately set the model parameters for a single target.

The core issue with any MHT algorithm is the exponential growth in the problem size. A novel approach to this could be to formulate a pruning score function for each node tree with the goal of removing nodes that does not add information while still having enough nodes to never render the original association optimization problem infeasible.

Chapter 7

Conclusion

To the knowledge of the author, this thesis has presented the first MHT implementation utilizing both AIS and radar information. It has shown the benefits of using AIS to aid an ILP based track oriented MHT integrated into a complete tracking system. The behaviour and performance for different tuning parameter in a logic based track initiator has also been demonstrated. Through simulations it has been shown that targets with low probability of detection benefits greatly from AIS aiding, whereas targets with moderate to high probability of detection gained less benefits.

Appendix A

Source code

pyMHT, the Python3 implementation of the MHT module developed in this thesis can be found at GitHub: https://github.com/erikliland/pyMHT.

Except for packages available via PyPI the only requirements is the Google Optimization Tools (OR-Tools) which can be obtained from https://developers.google.com/optimization/, and Jacob Frelinger's Cython / C++ implementation of the Munkres algorithm https://github.com/jfrelinger/cython-munkres-wrapper. The later is installed automatically with the pyMHT package.

Appendix B

Initialization time plot

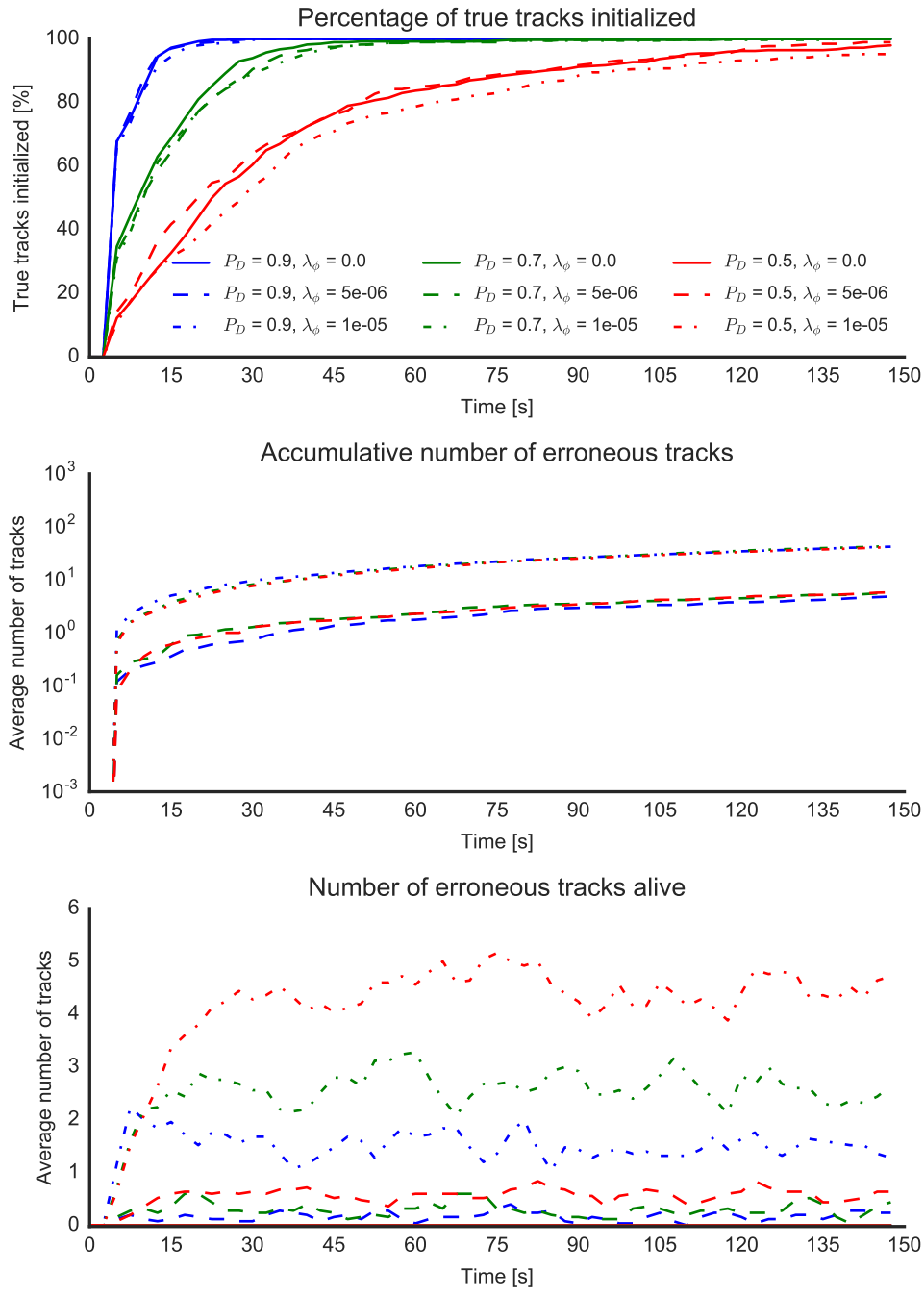

Figure B.1: Scenario 0 — Initialization time (1/1)

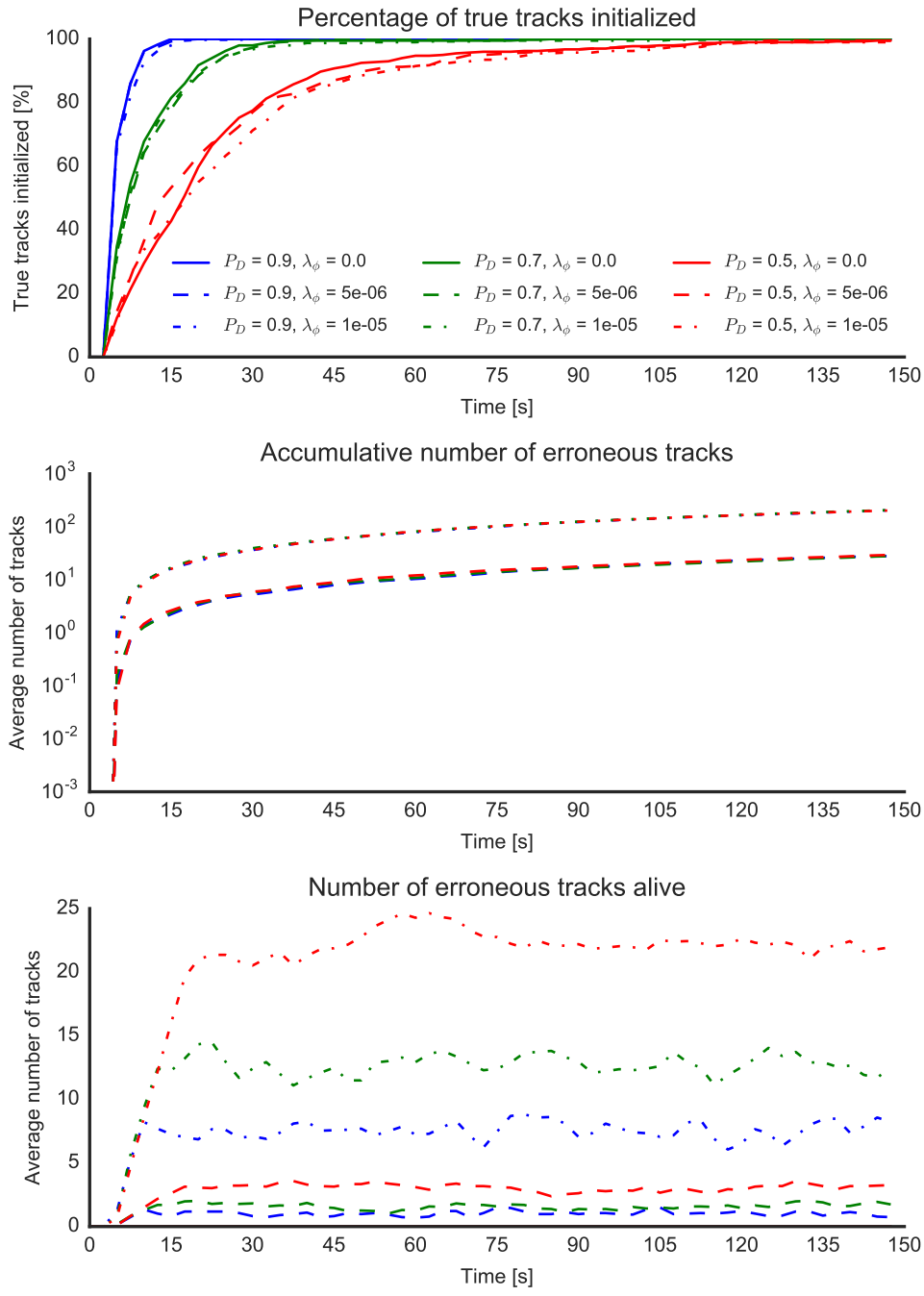

Figure B.2: Scenario 0 — Initialization time (1/2)

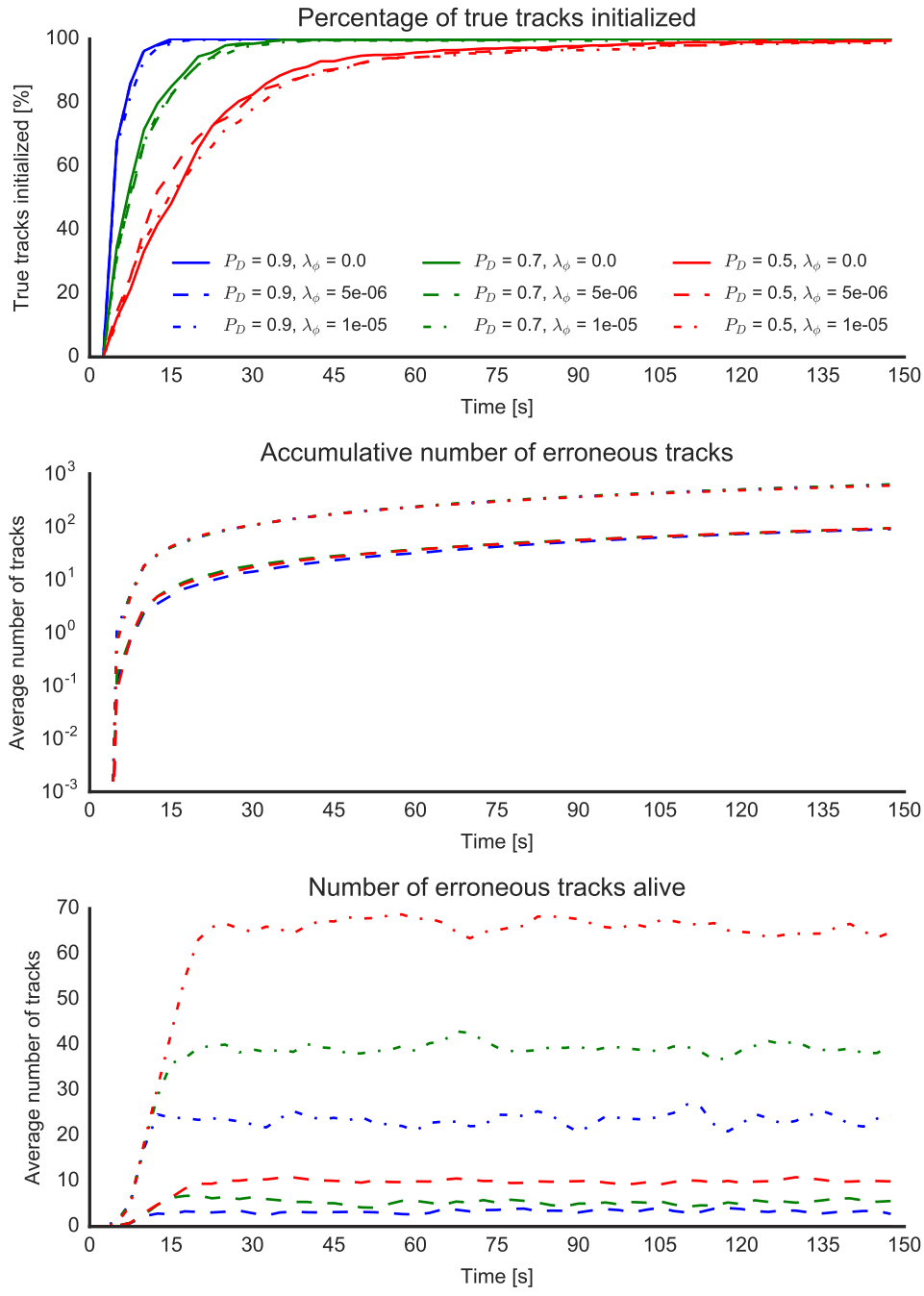

Figure B.3: Scenario 0 — Initialization time (1/3)

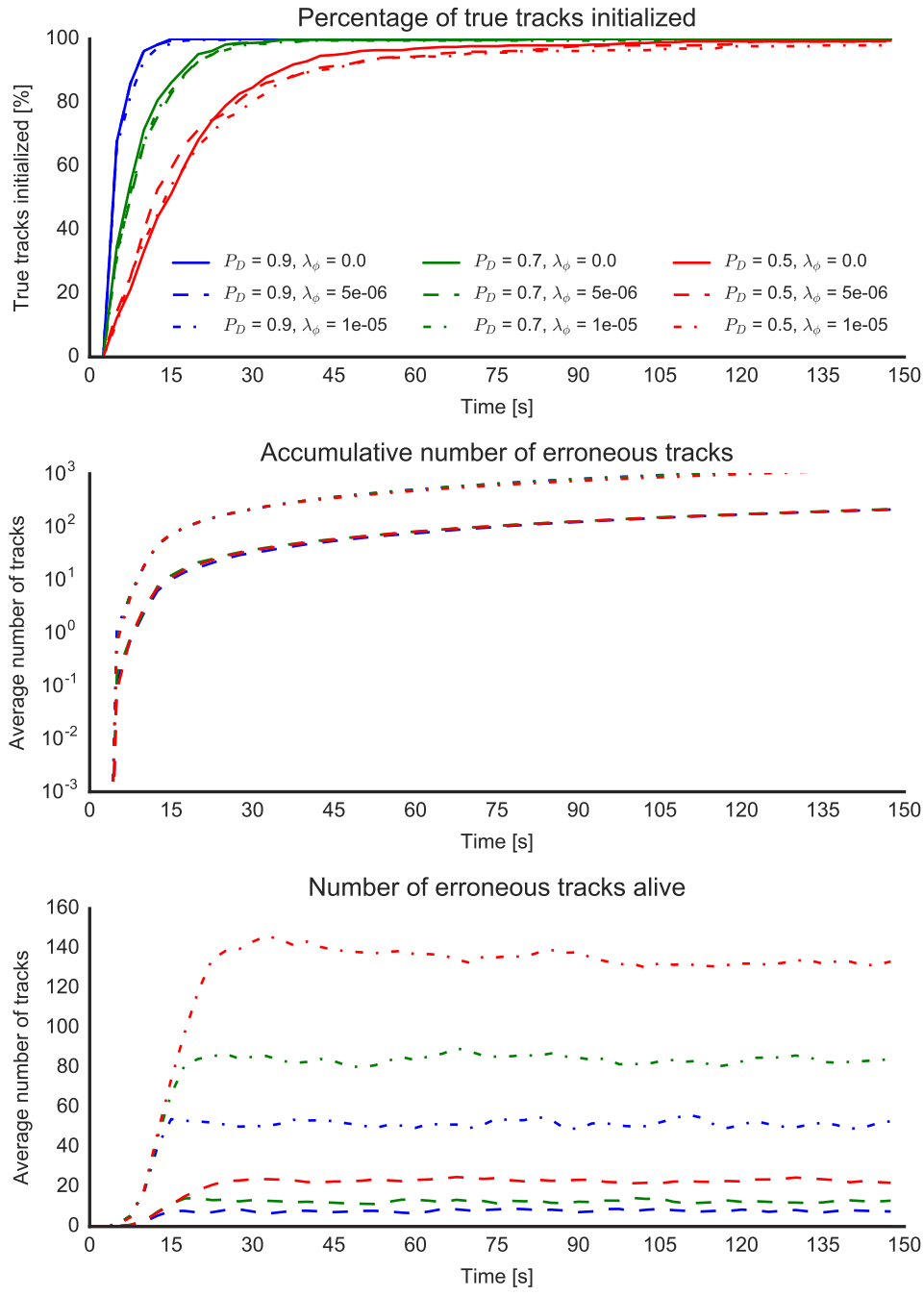

Figure B.4: Scenario 0 — Initialization time (1/4)

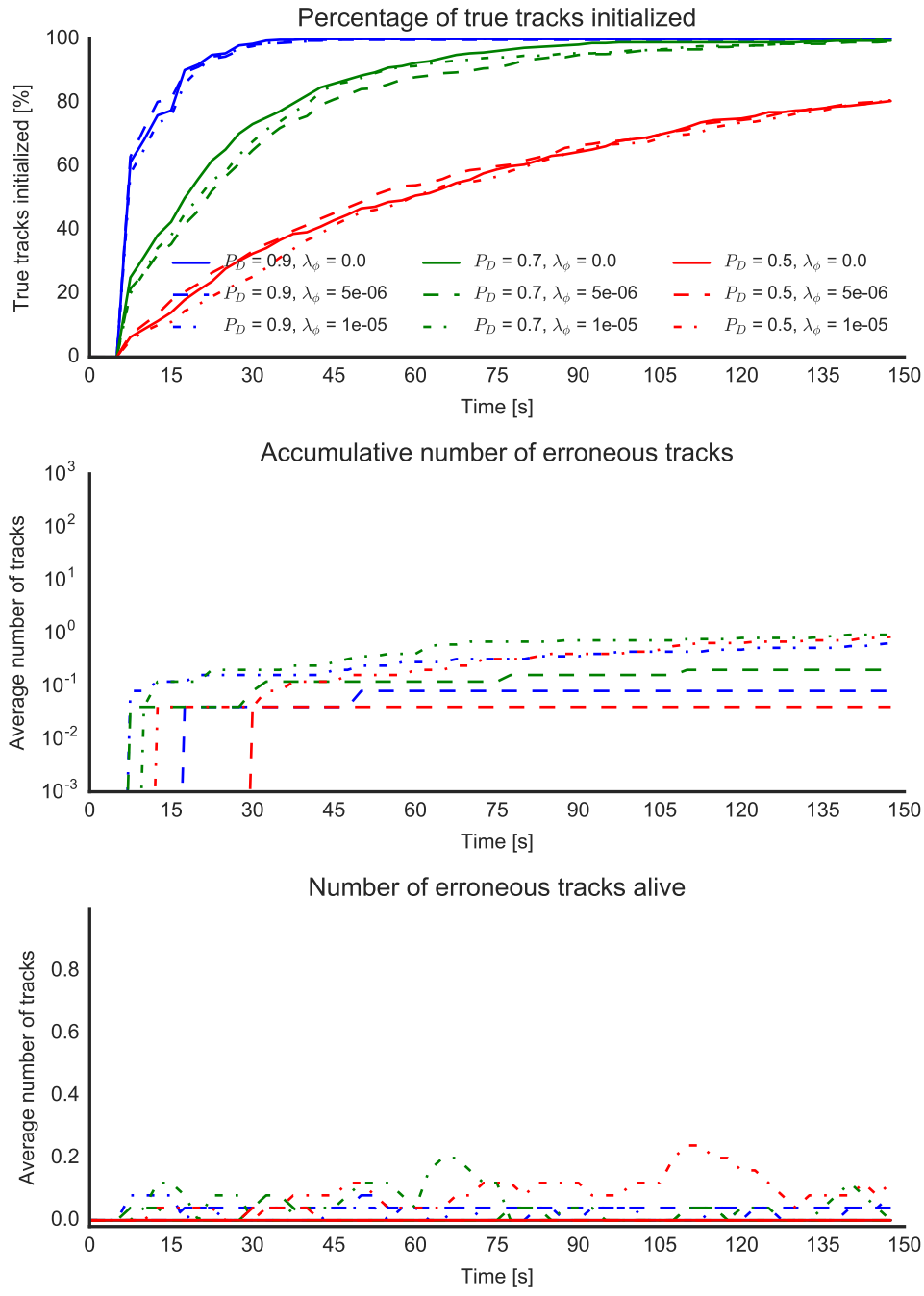

Figure B.5: Scenario 0 — Initialization time (2/2)

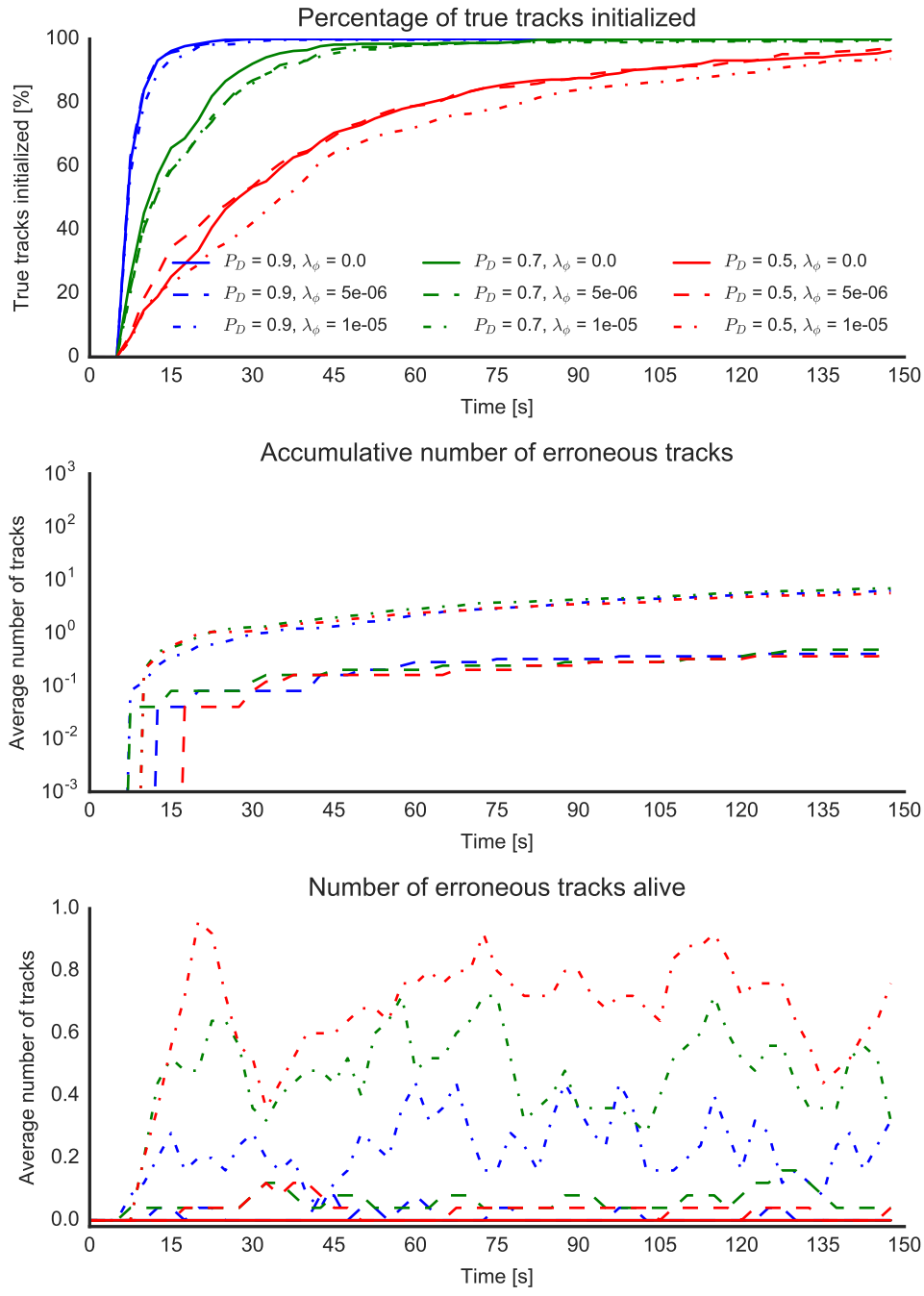

Figure B.6: Scenario 0 — Initialization time (2/3)

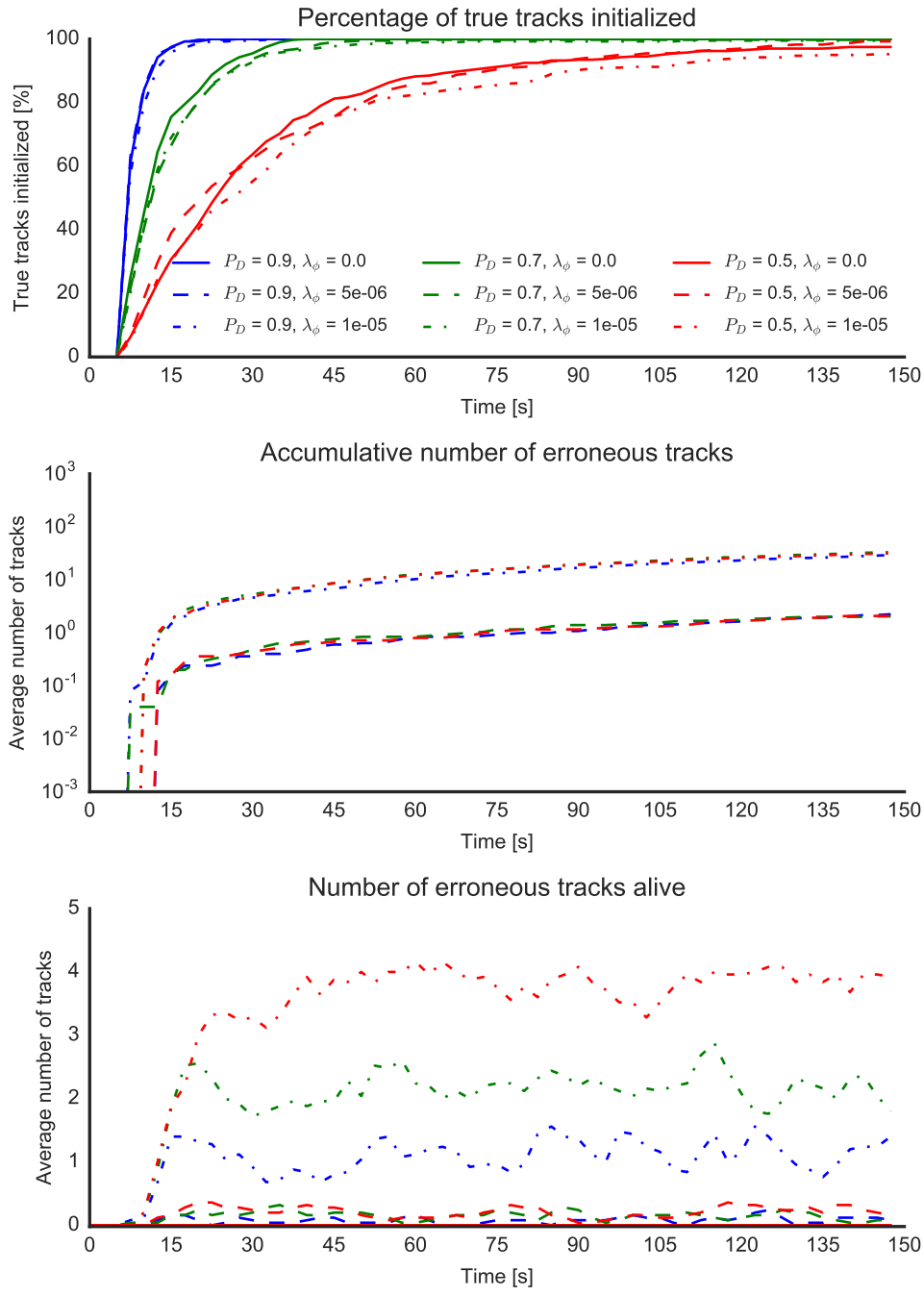

Figure B.7: Scenario 0 — Initialization time (2/4)

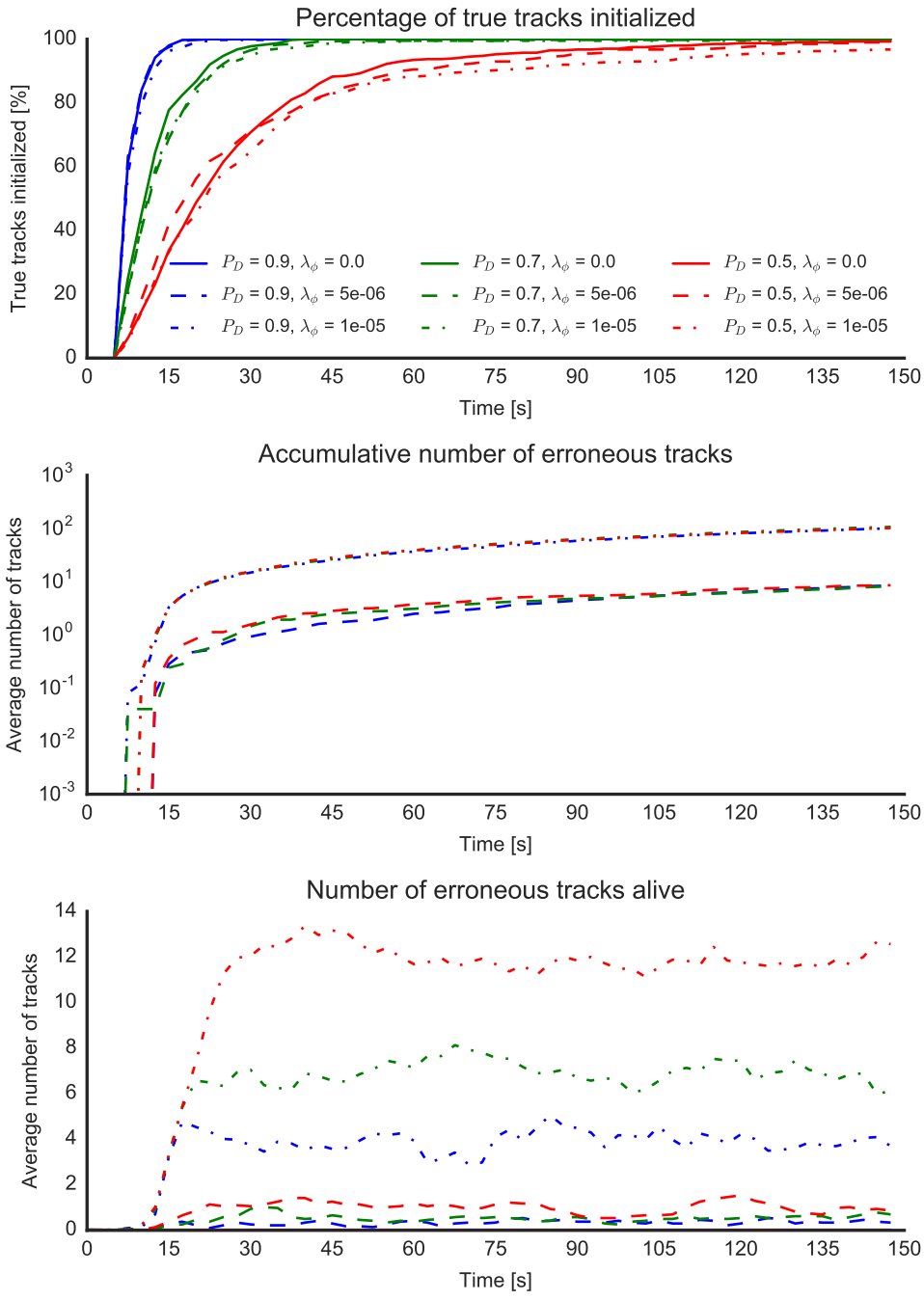

Figure B.8: Scenario 0 — Initialization time (2/5)

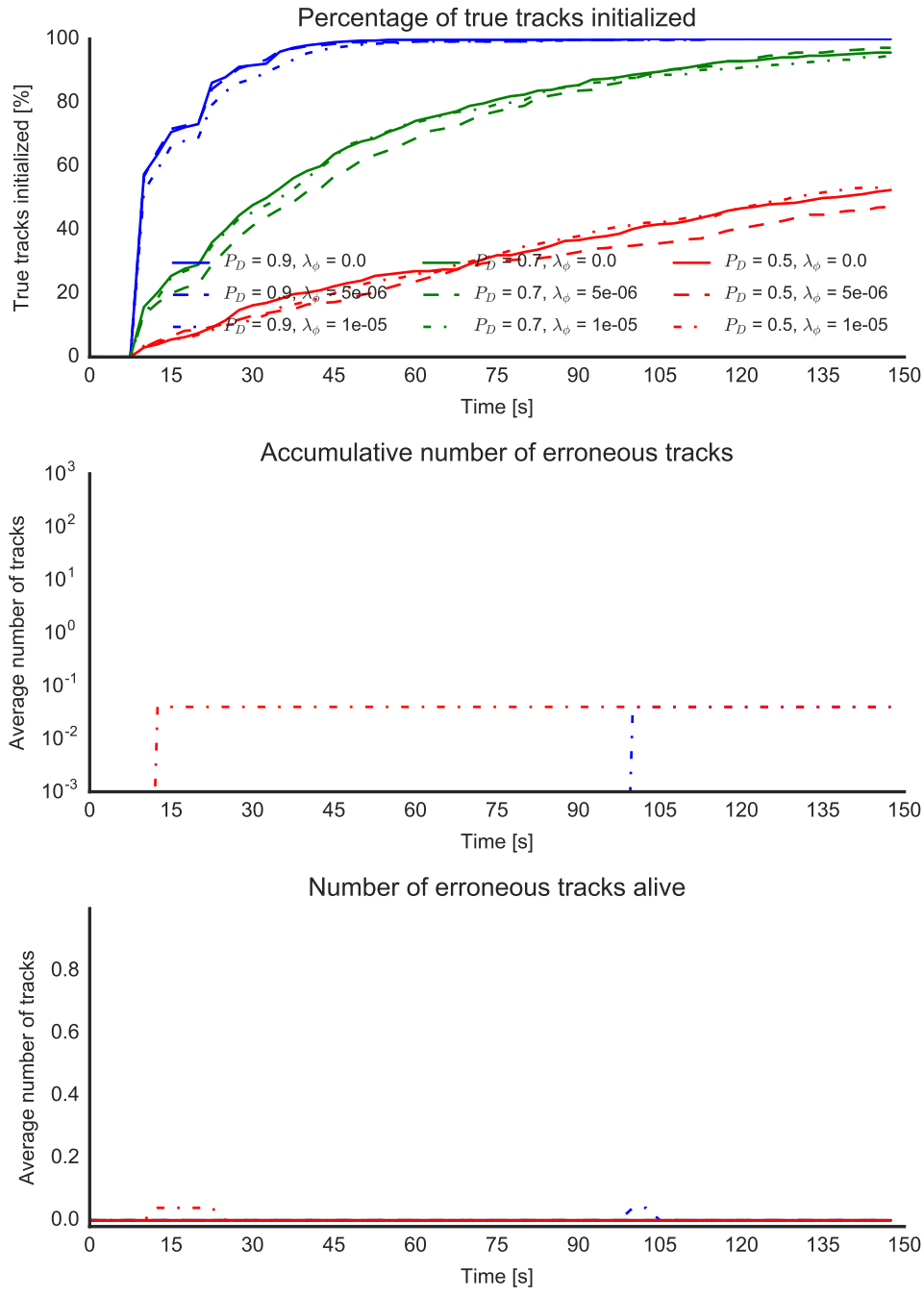

Figure B.9: Scenario 0 — Initialization time (3/3)

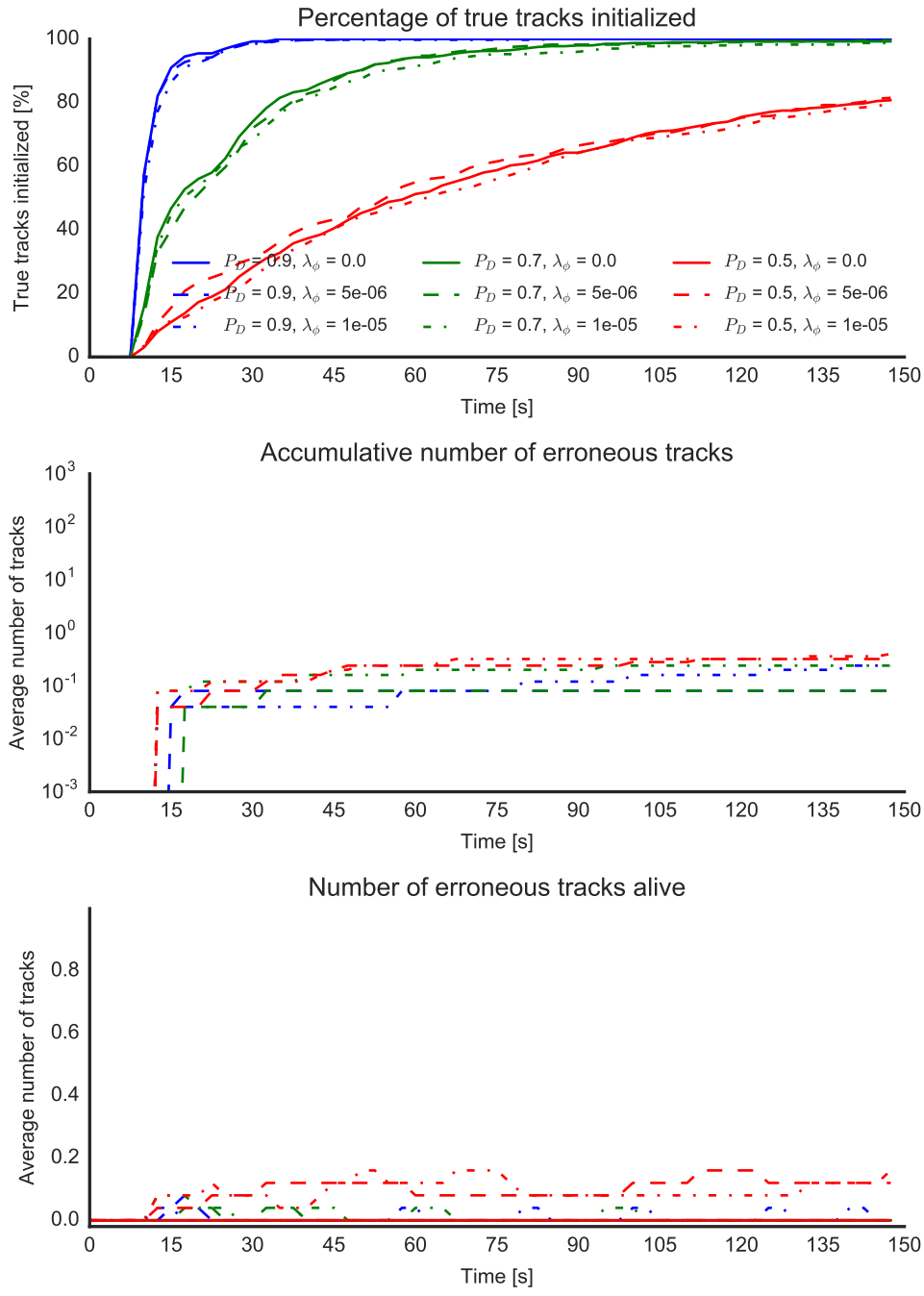

Figure B.10: Scenario 0 — Initialization time (3/4)

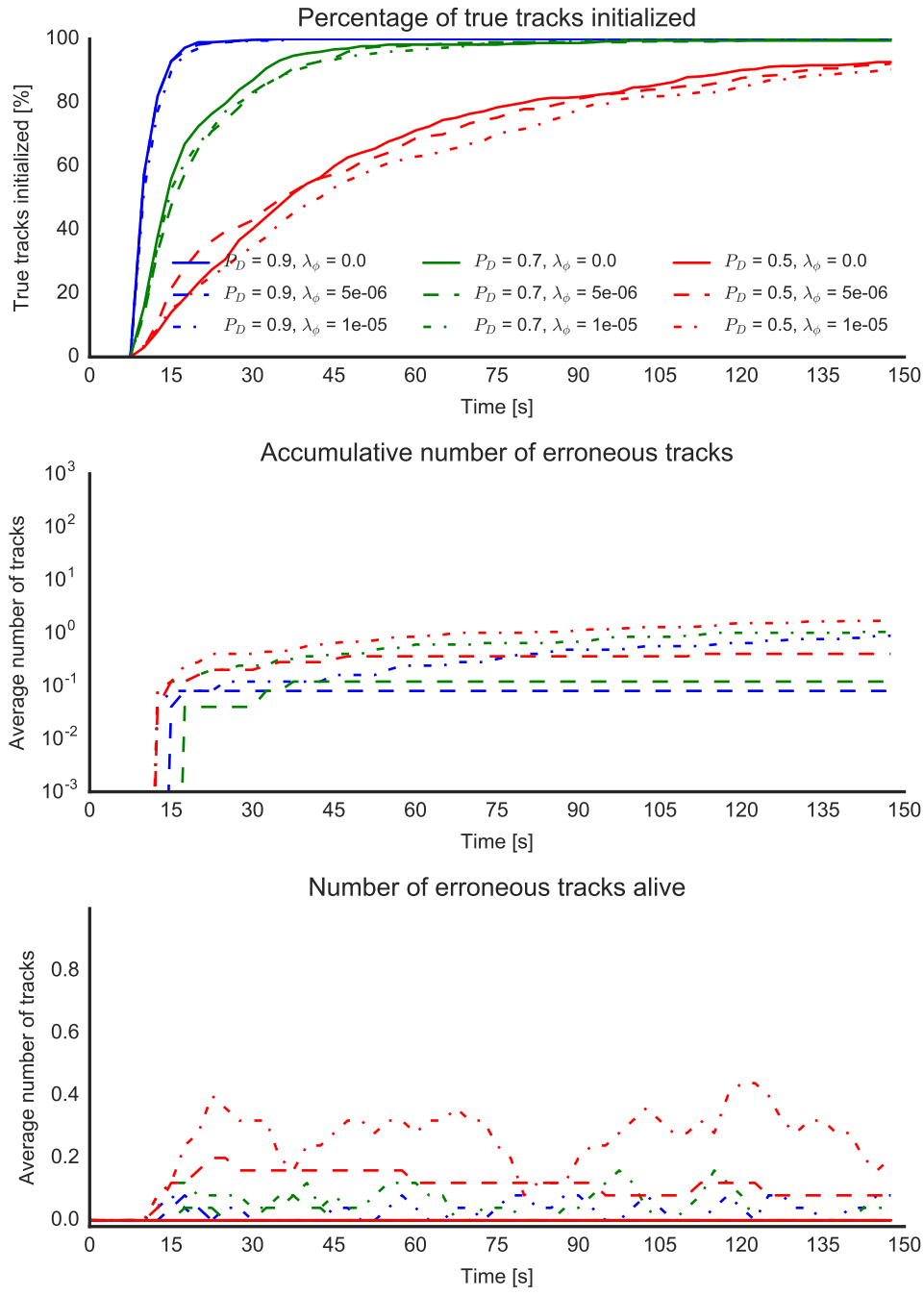

Figure B.11: Scenario 0 — Initialization time (3/5)

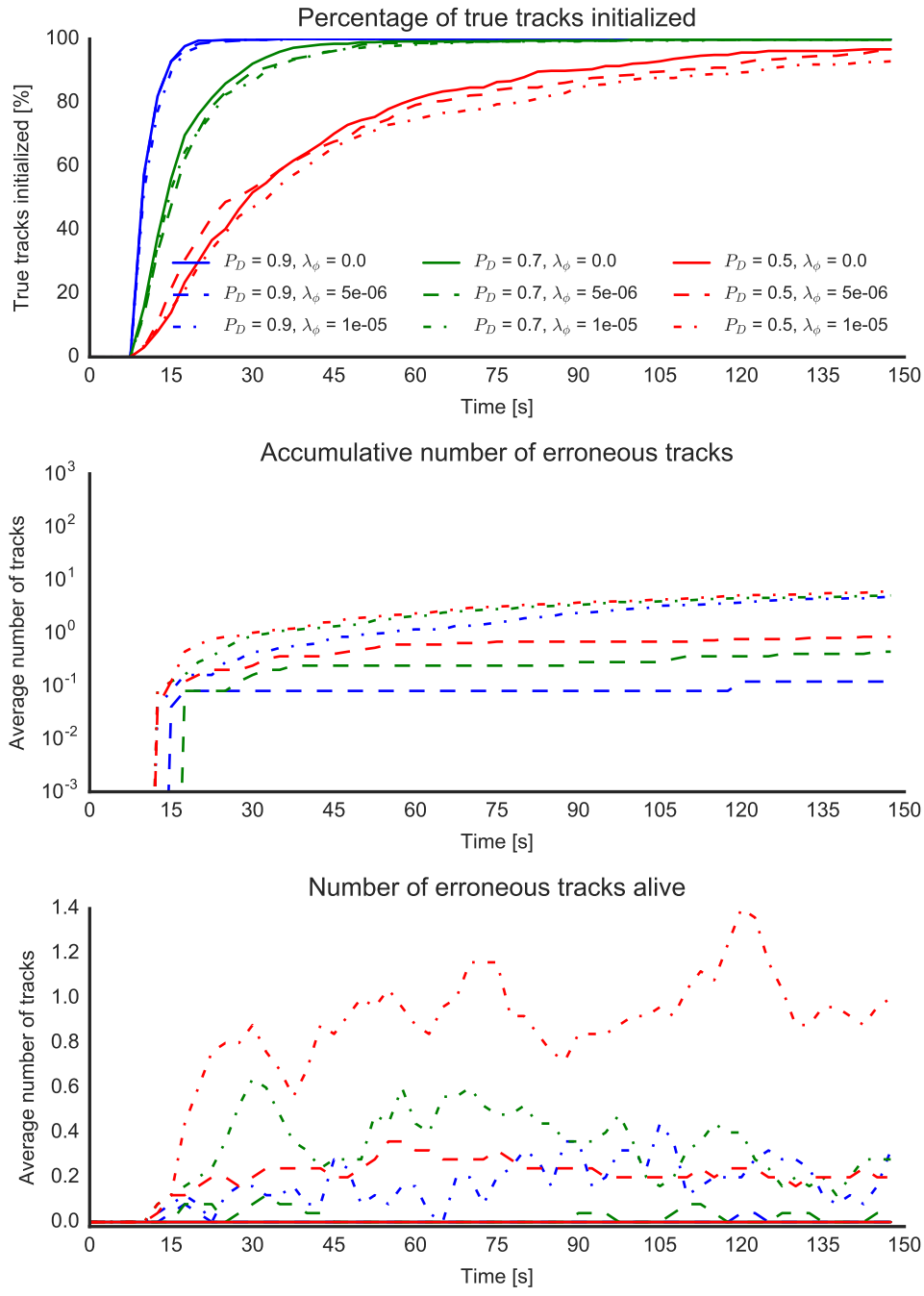

Figure B.12: Scenario 0 — Initialization time (3/6)

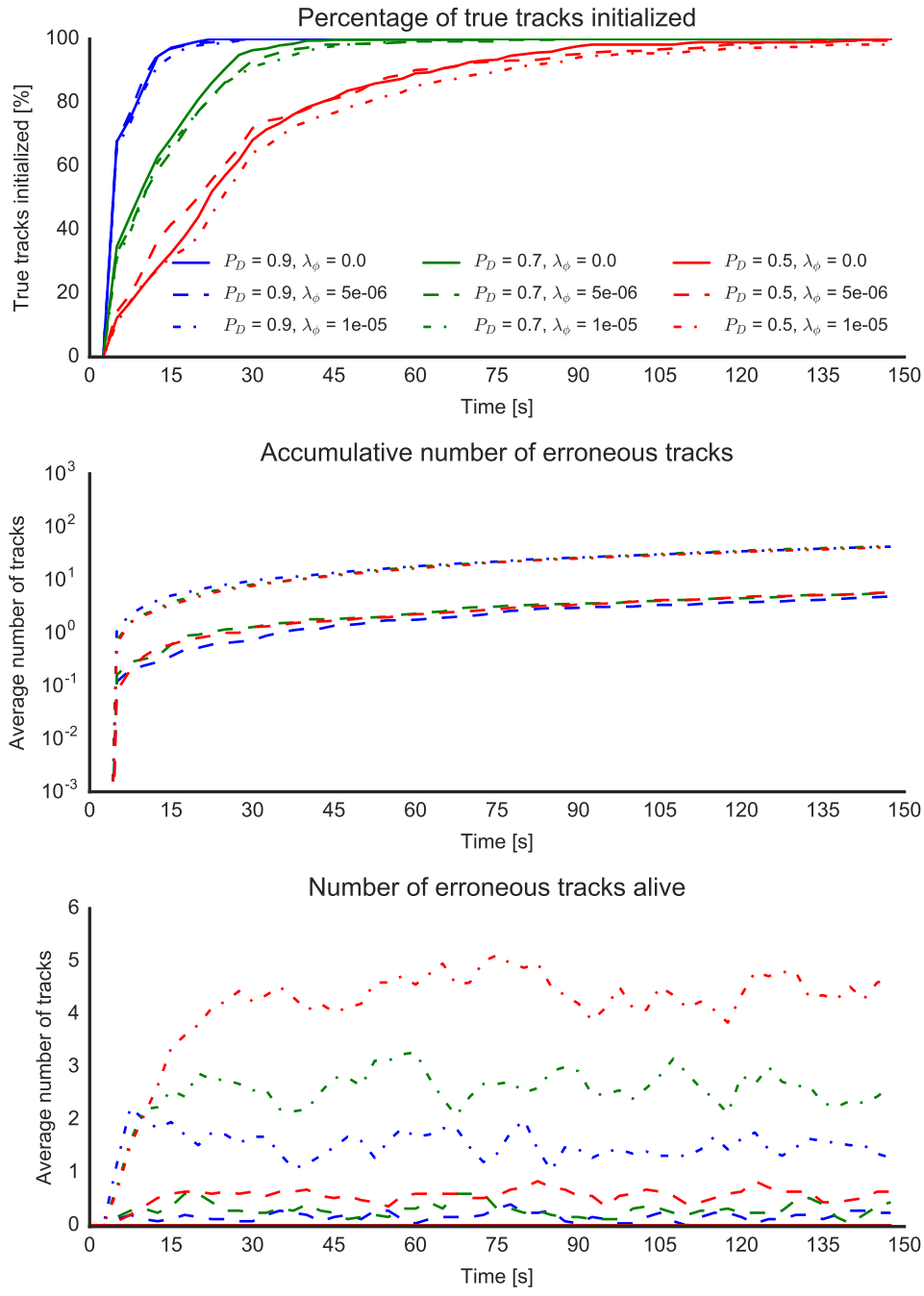

Figure B.13: Scenario 1 — Initialization time (1/1)

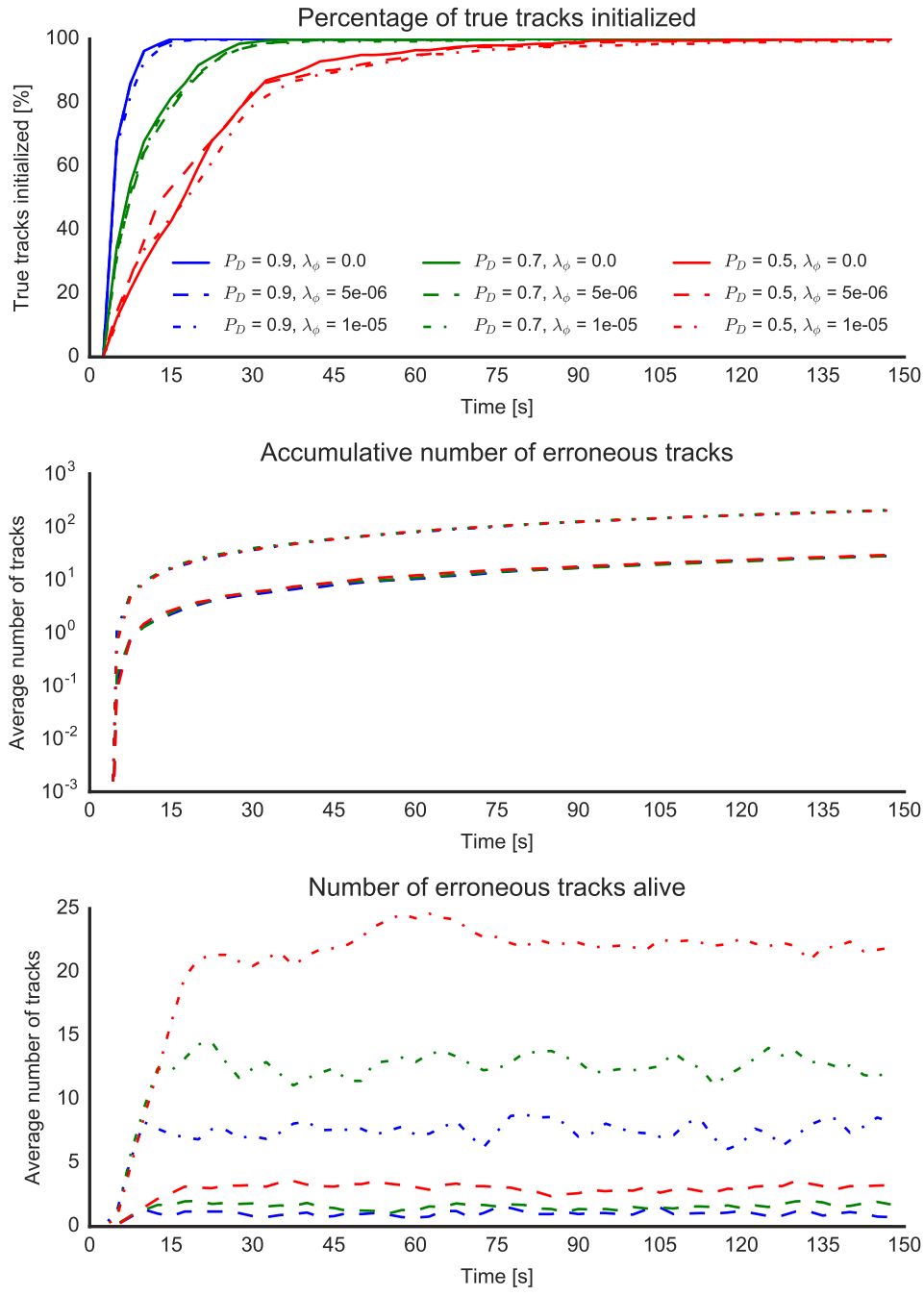

Figure B.14: Scenario 1 — Initialization time (1/2)

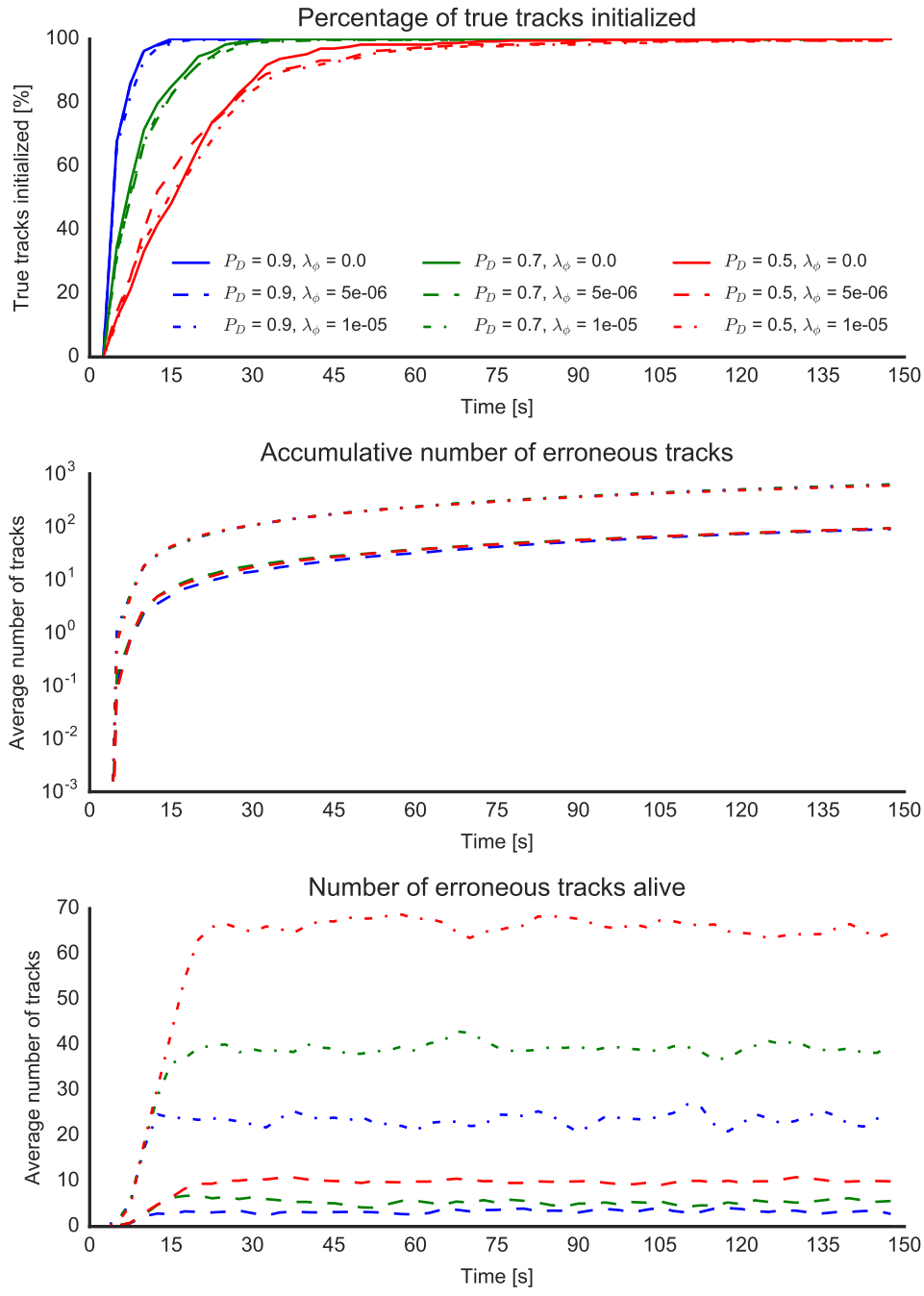

Figure B.15: Scenario 1 — Initialization time (1/3)

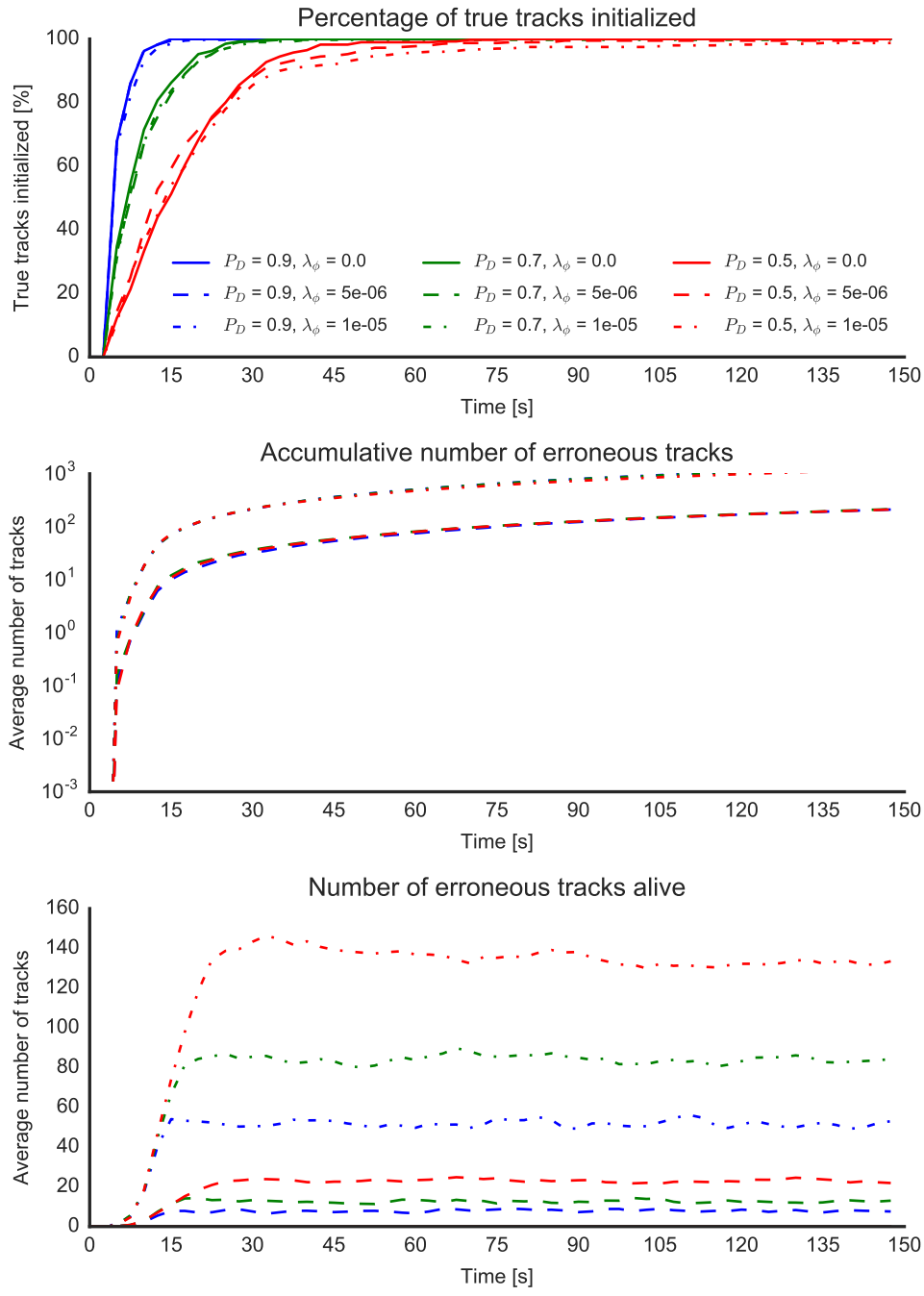

Figure B.16: Scenario 1 — Initialization time (1/4)

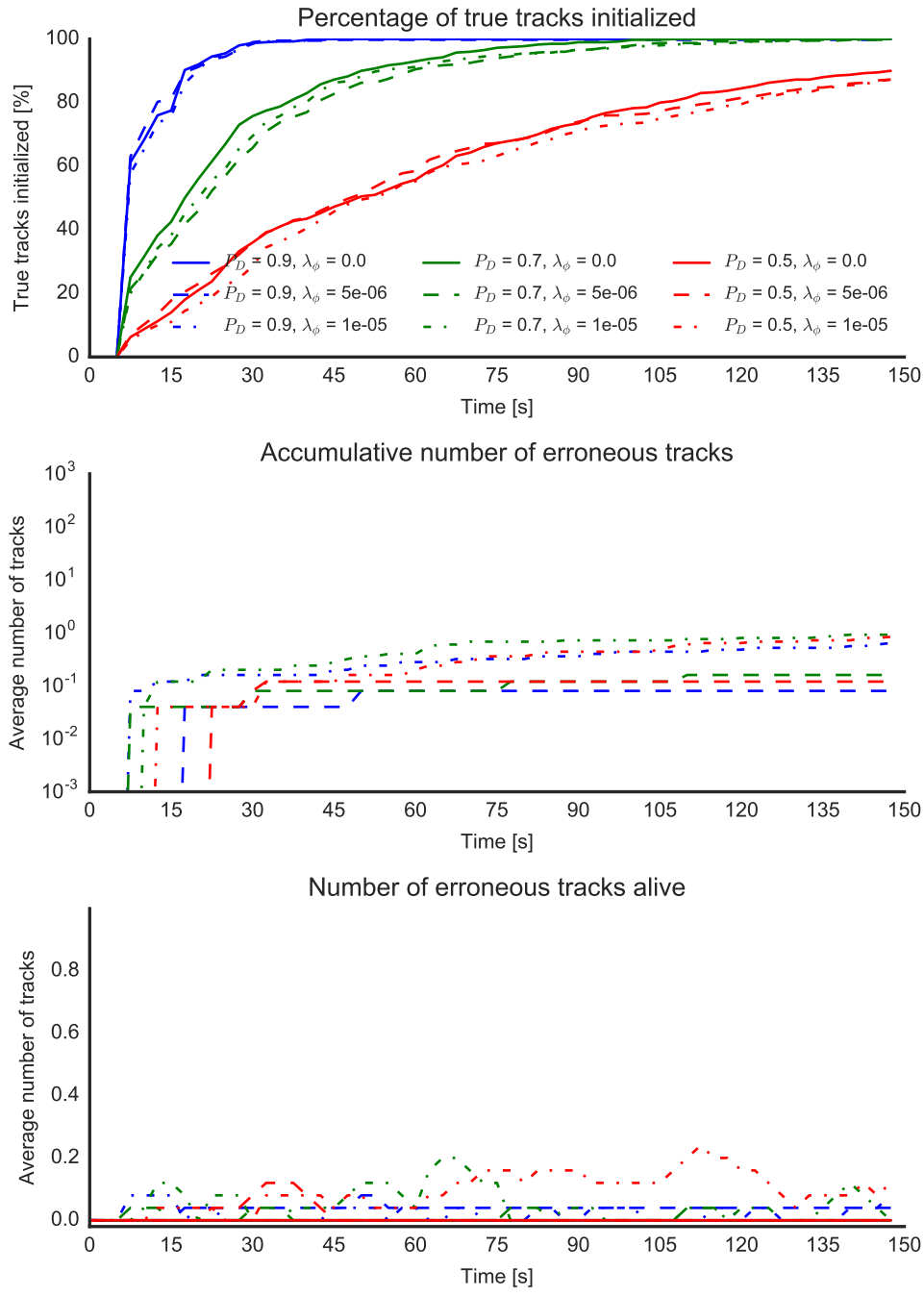

Figure B.17: Scenario 1 — Initialization time (2/2)

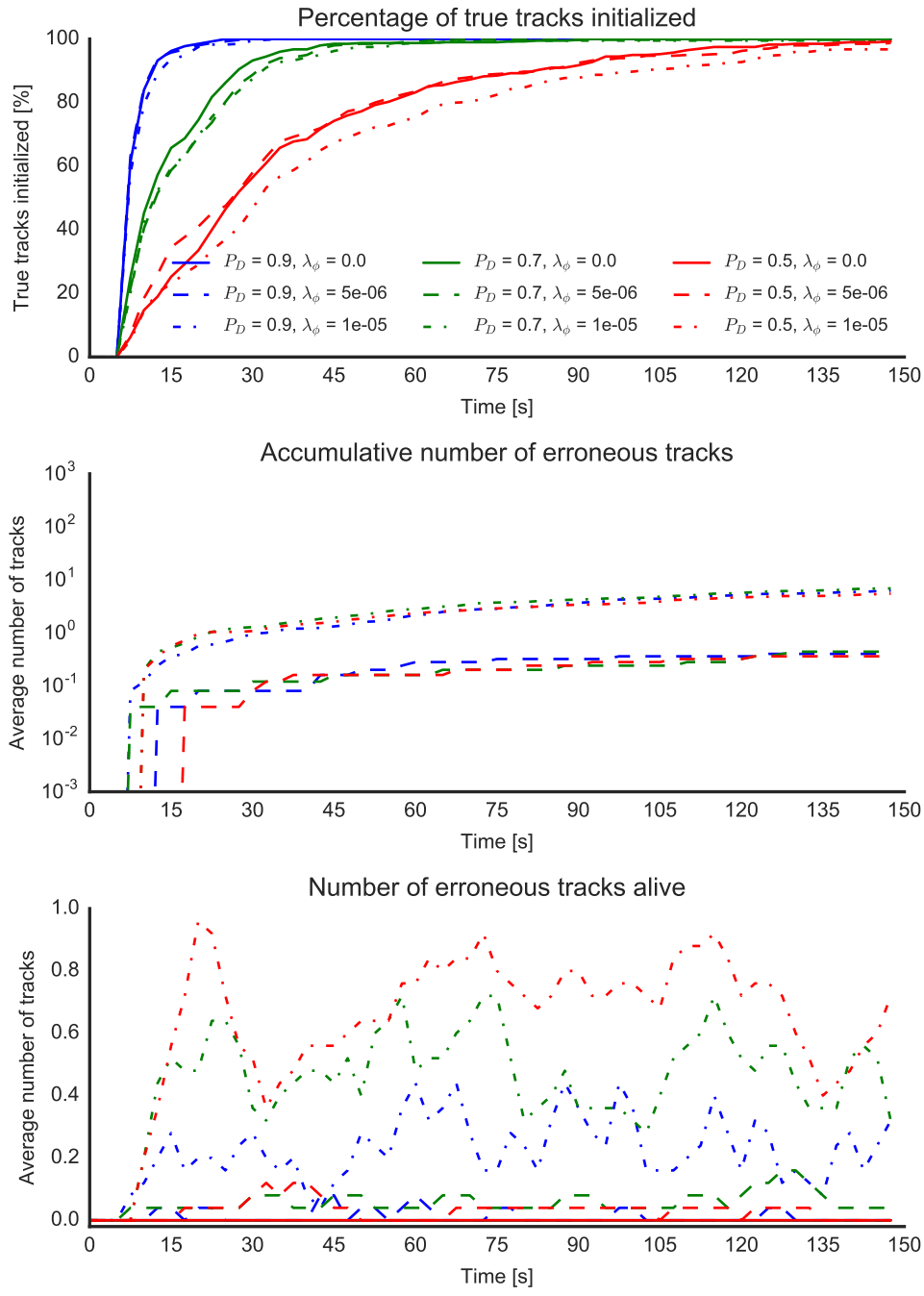

Figure B.18: Scenario 1 — Initialization time (2/3)

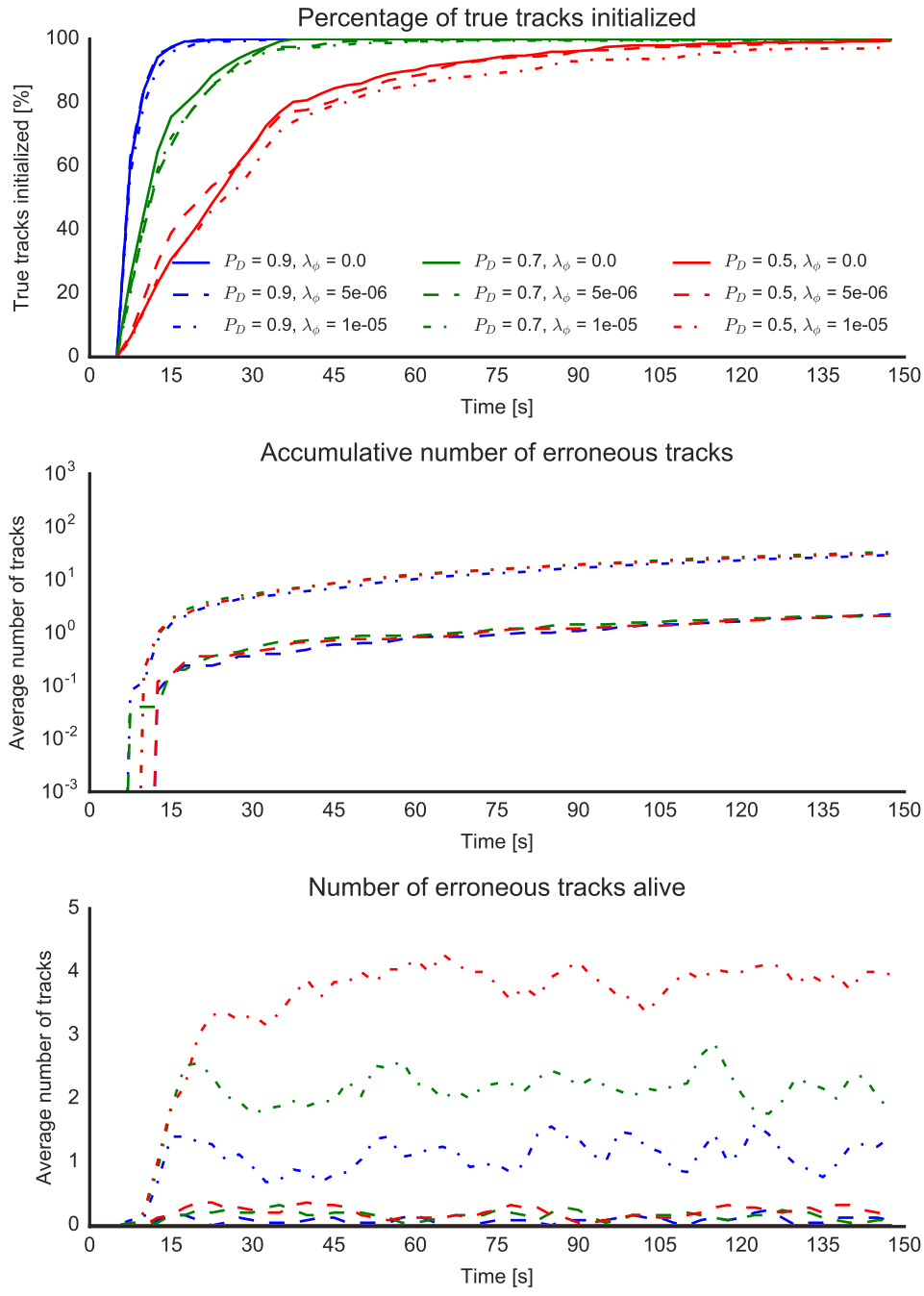

Figure B.19: Scenario 1 — Initialization time (2/4)

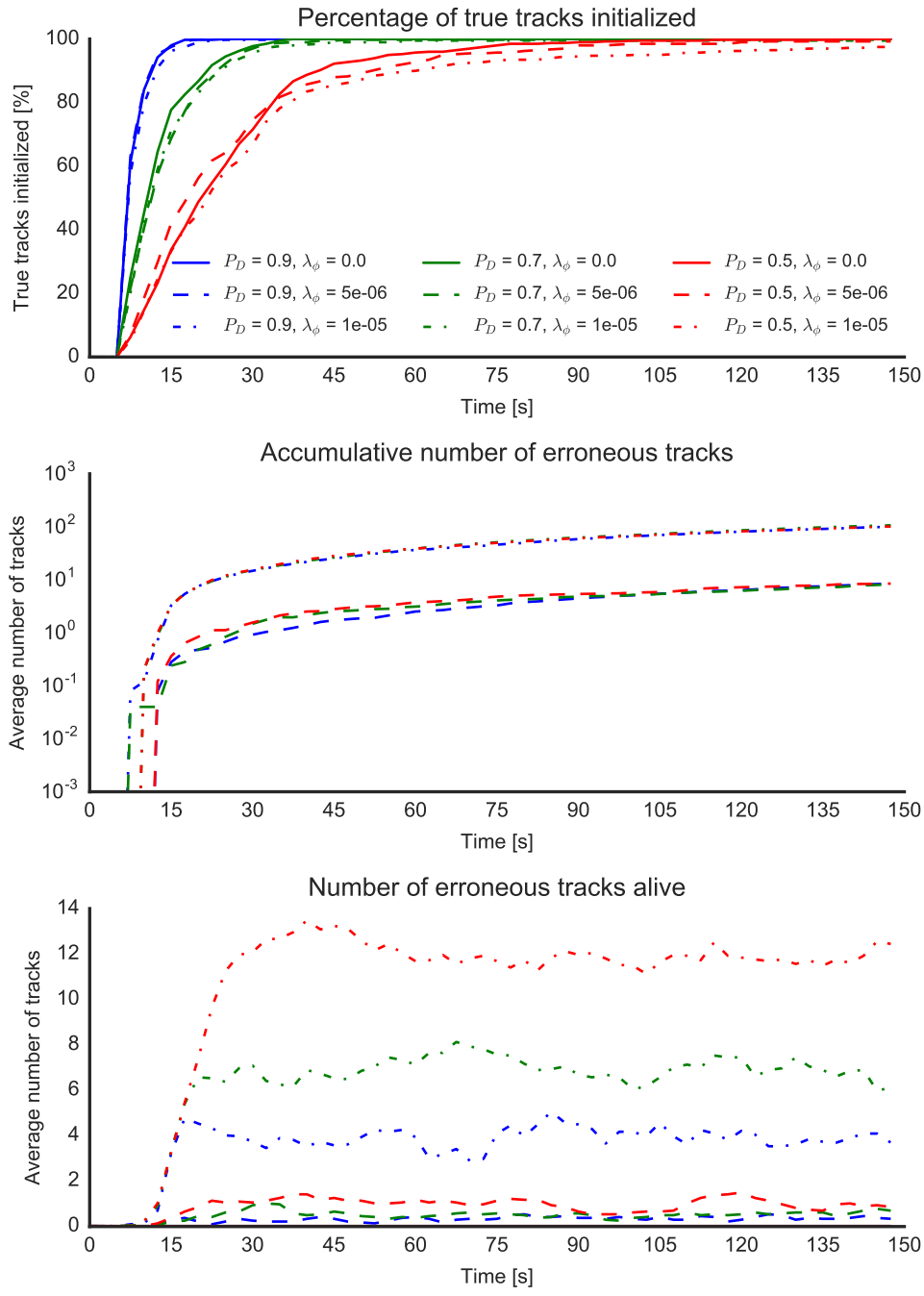

Figure B.20: Scenario 1 — Initialization time (2/5)

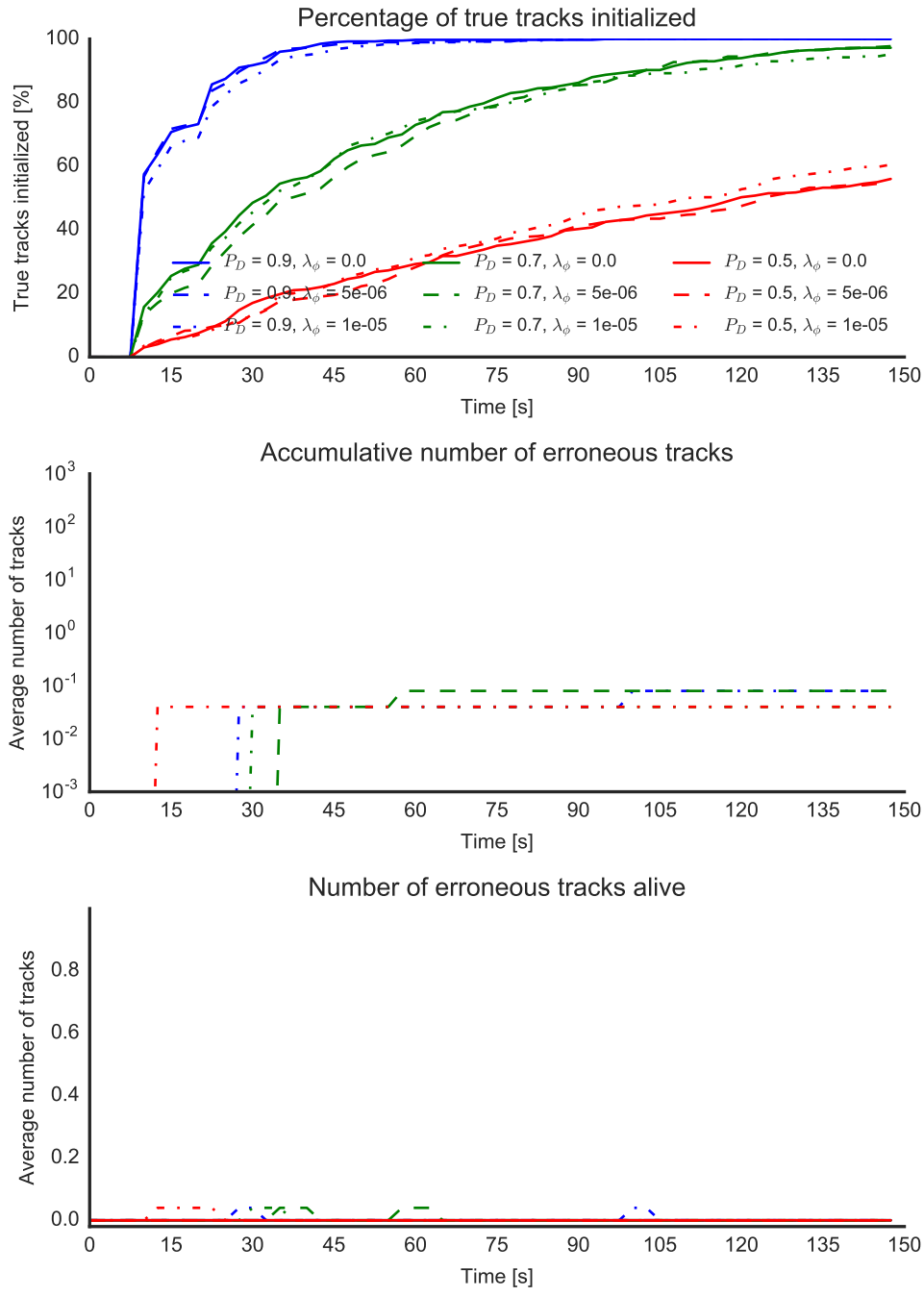

Figure B.21: Scenario 1 — Initialization time (3/3)

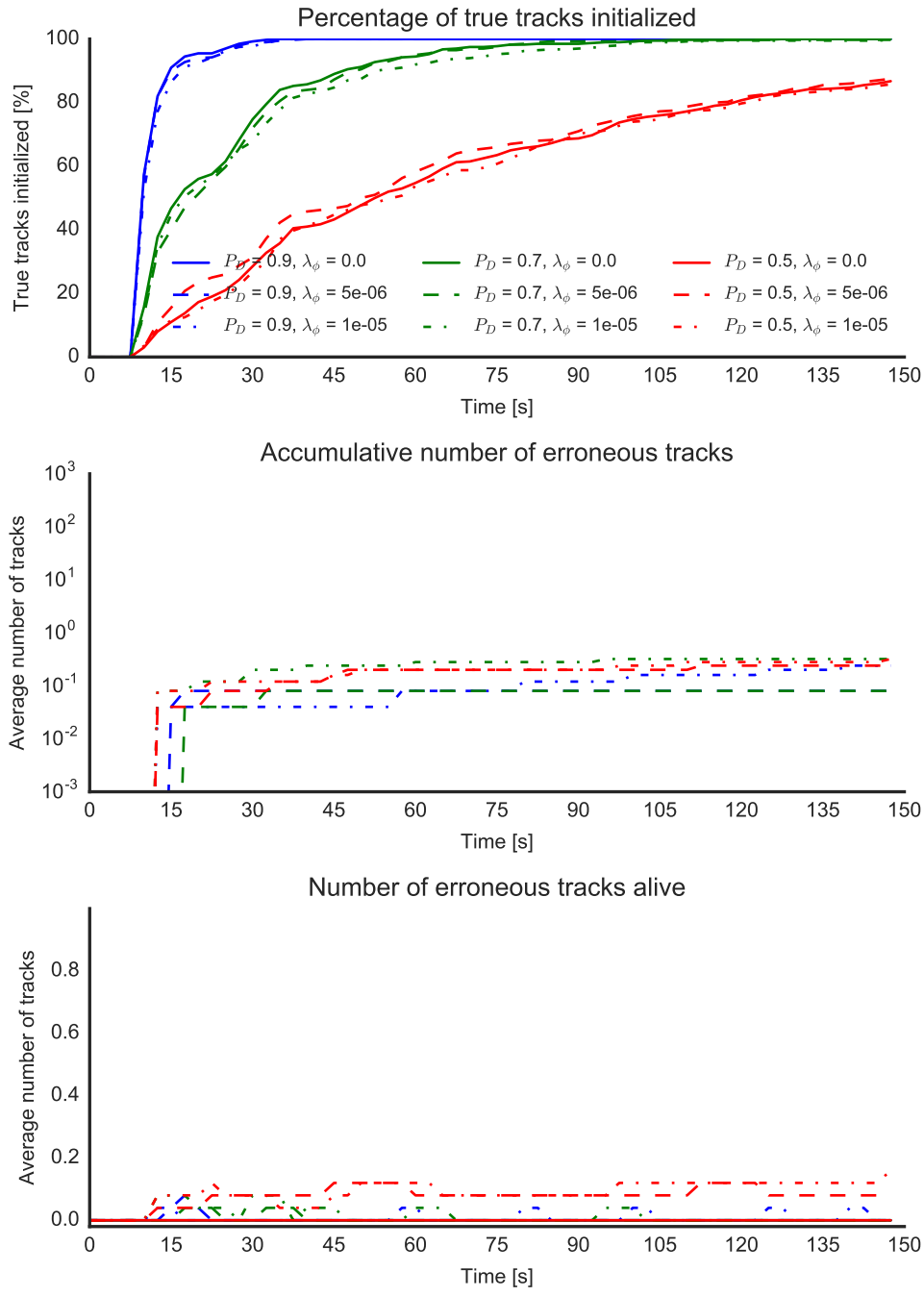

Figure B.22: Scenario 1 — Initialization time (3/4)

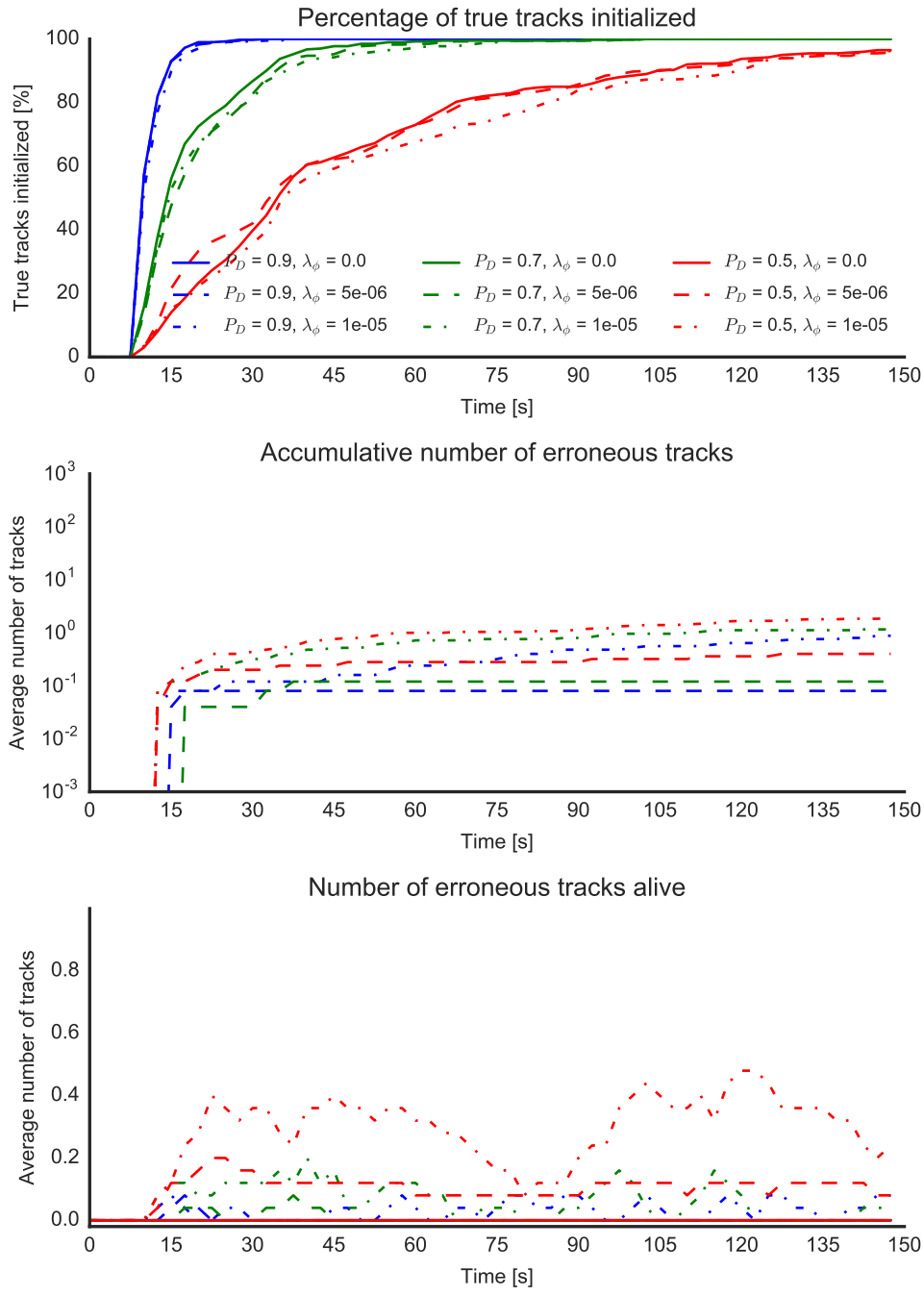

Figure B.23: Scenario 1 — Initialization time (3/5)

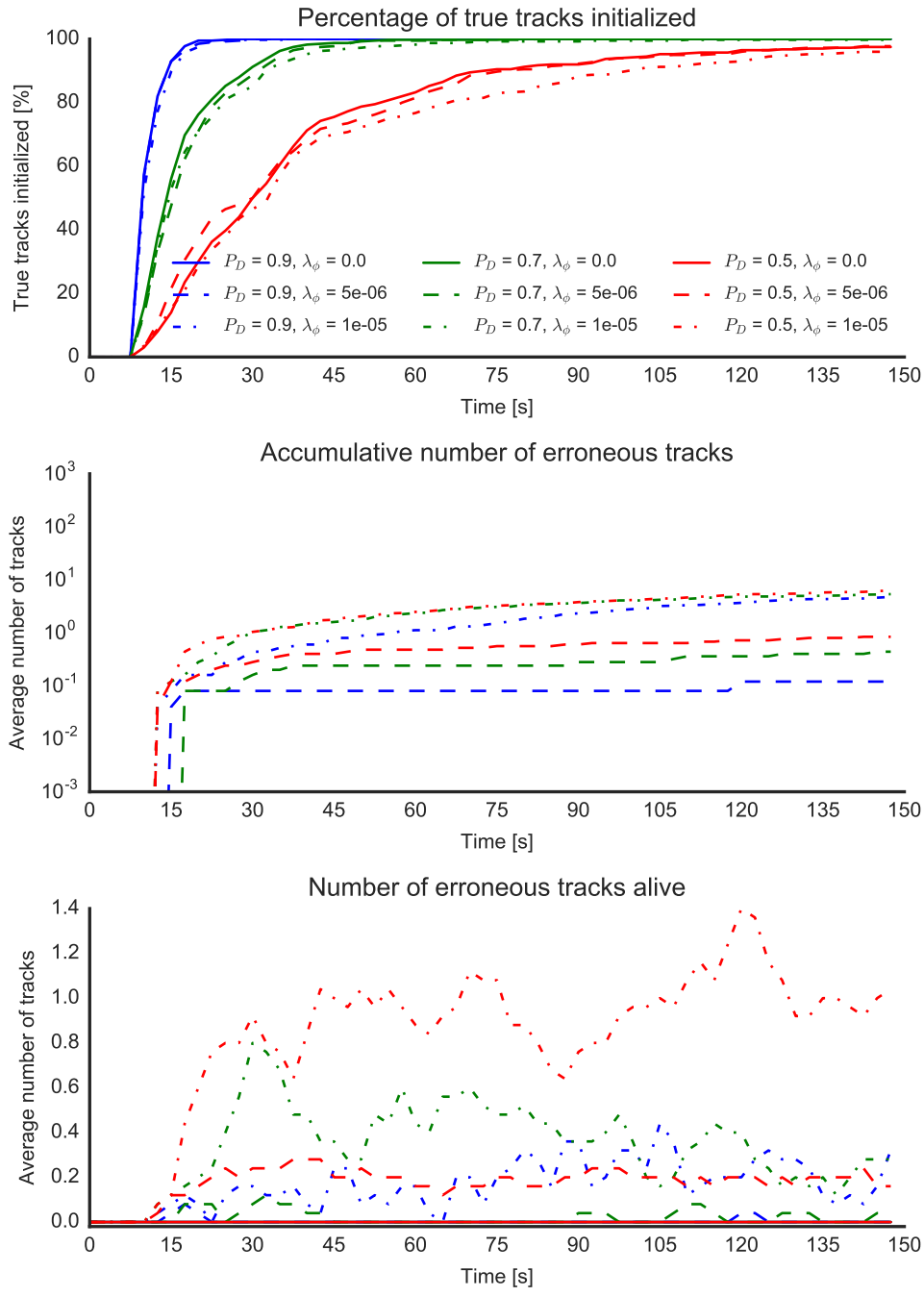

Figure B.24: Scenario 1 — Initialization time (3/6)

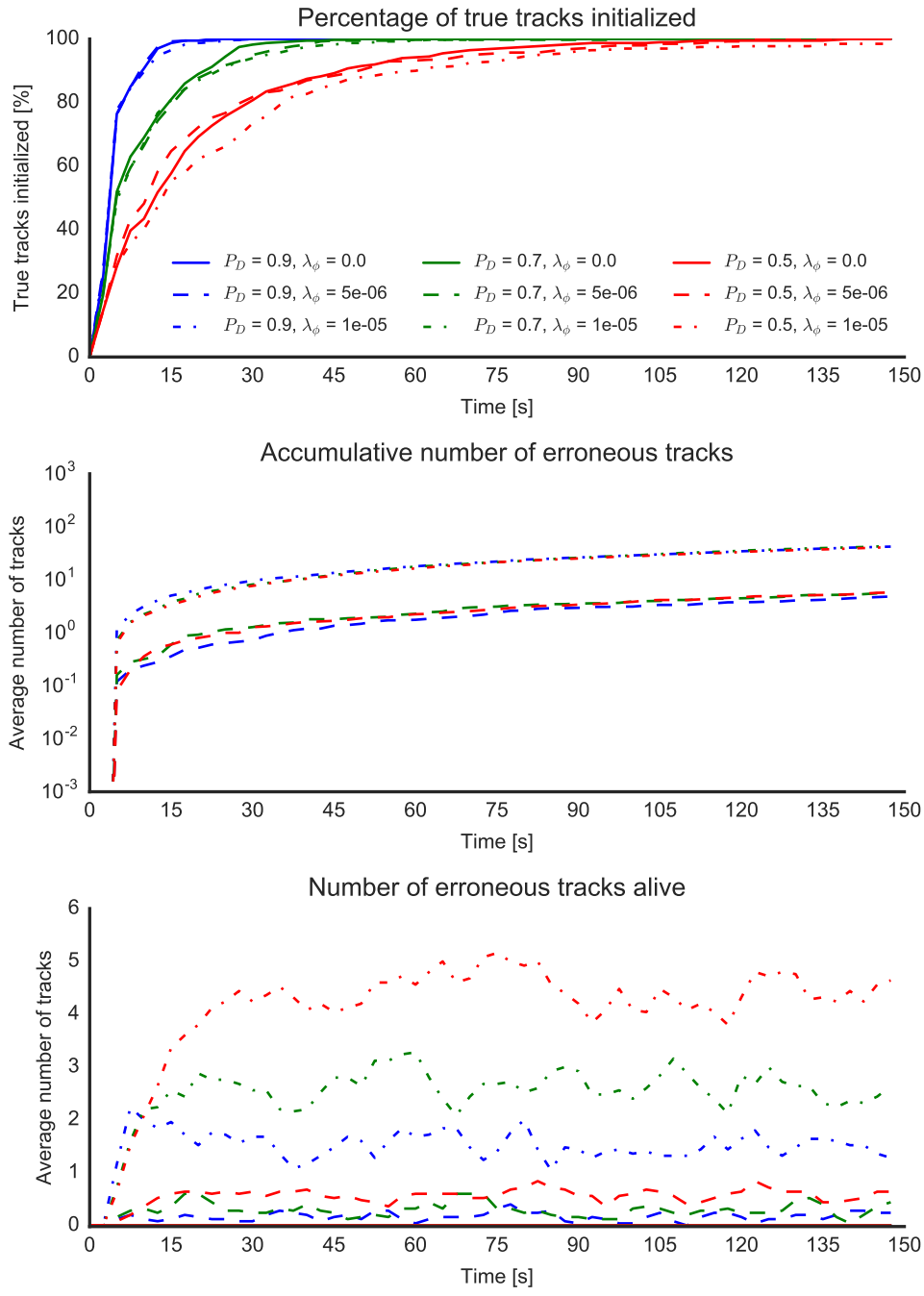

Figure B.25: Scenario 2 — Initialization time (1/1)

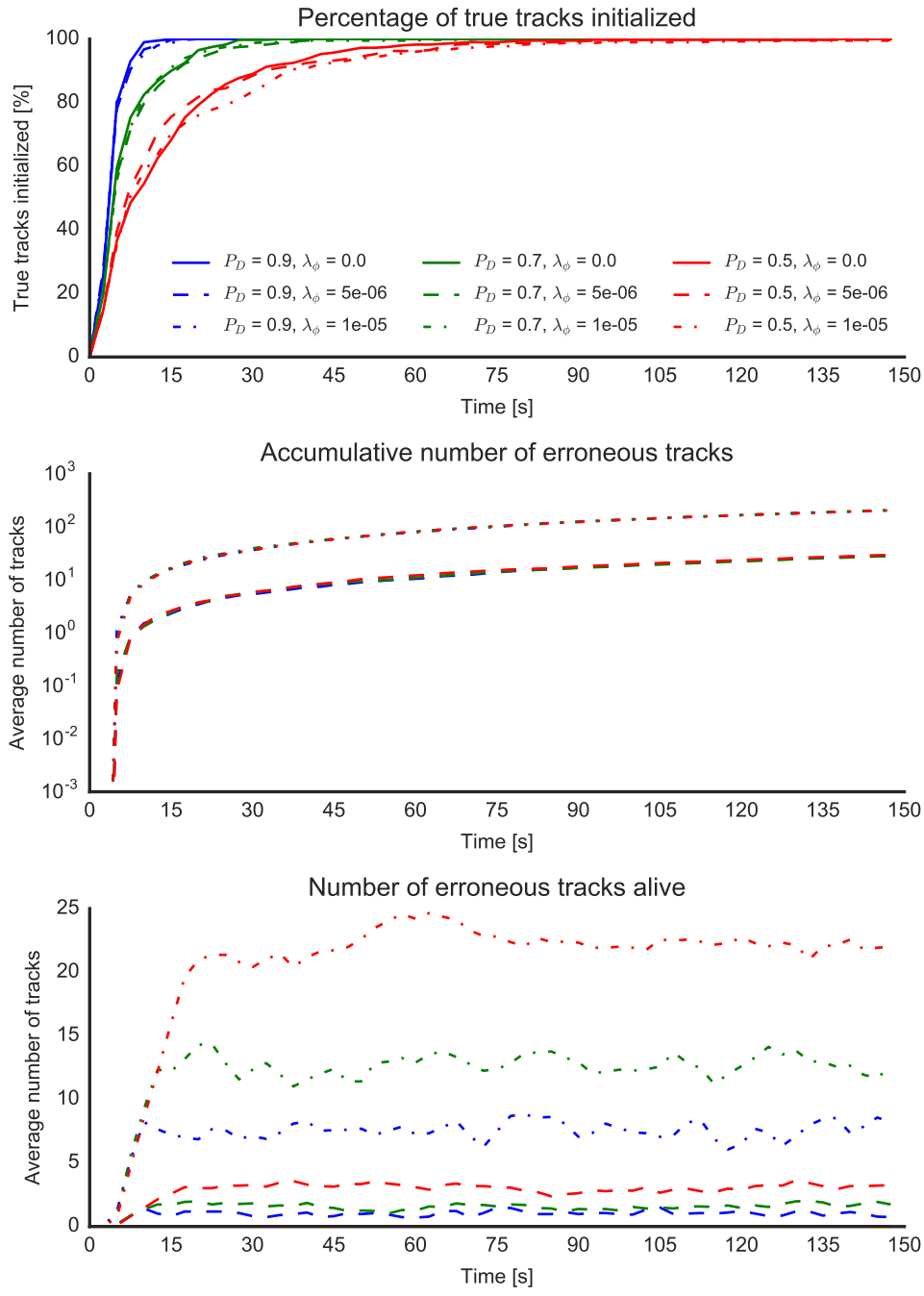

Figure B.26: Scenario 2 — Initialization time (1/2)

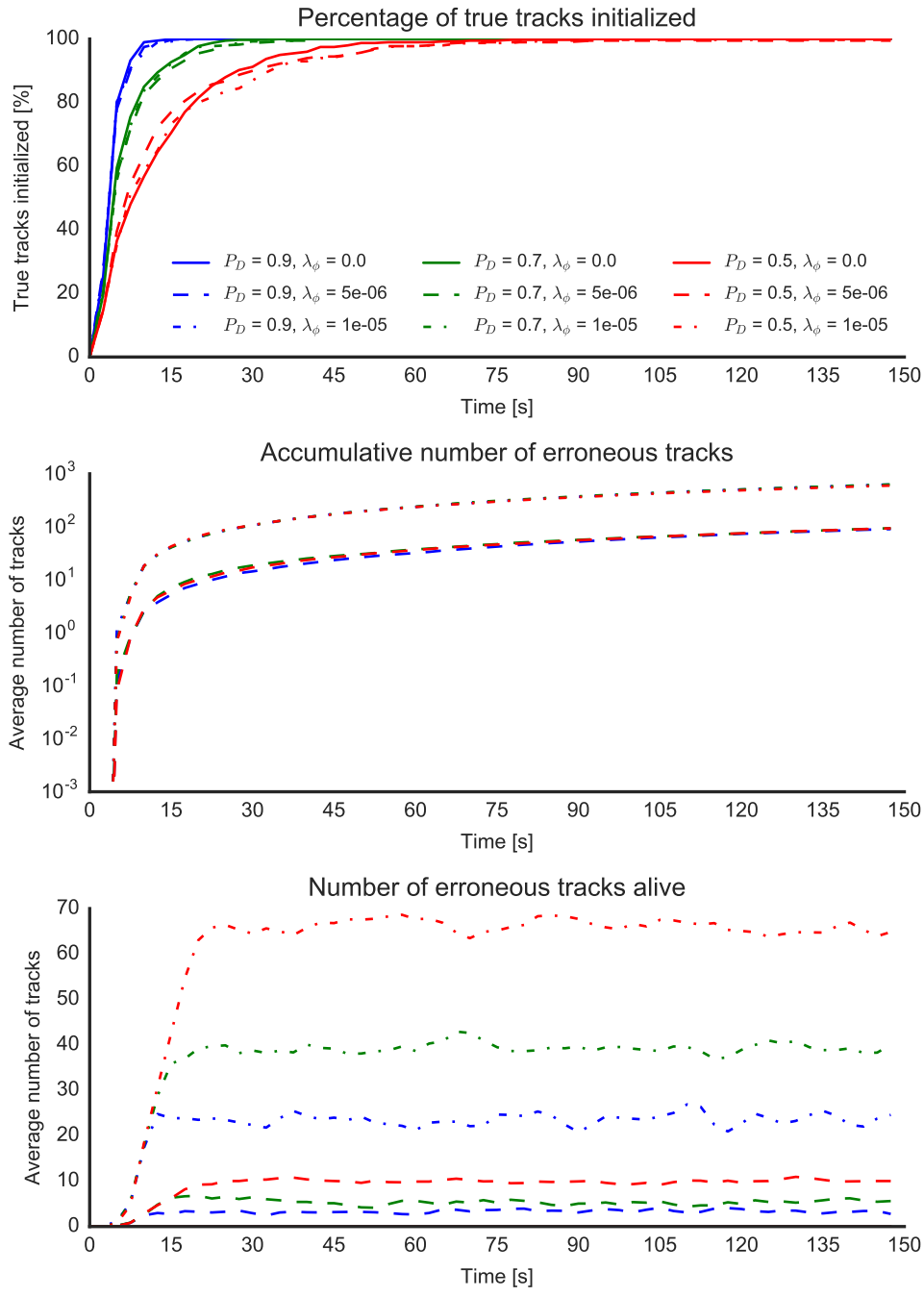

Figure B.27: Scenario 2 — Initialization time (1/3)

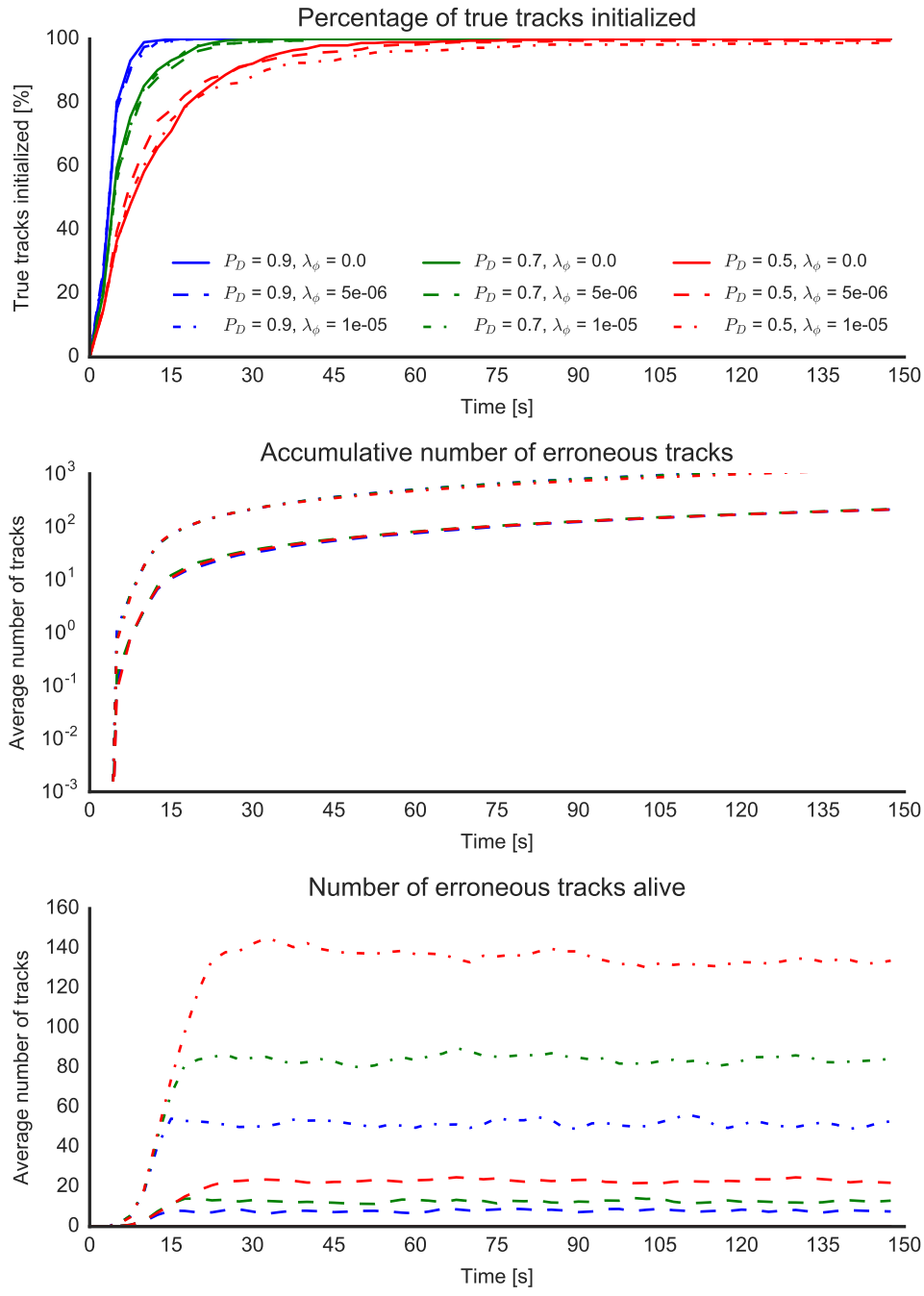

Figure B.28: Scenario 2 — Initialization time (1/4)

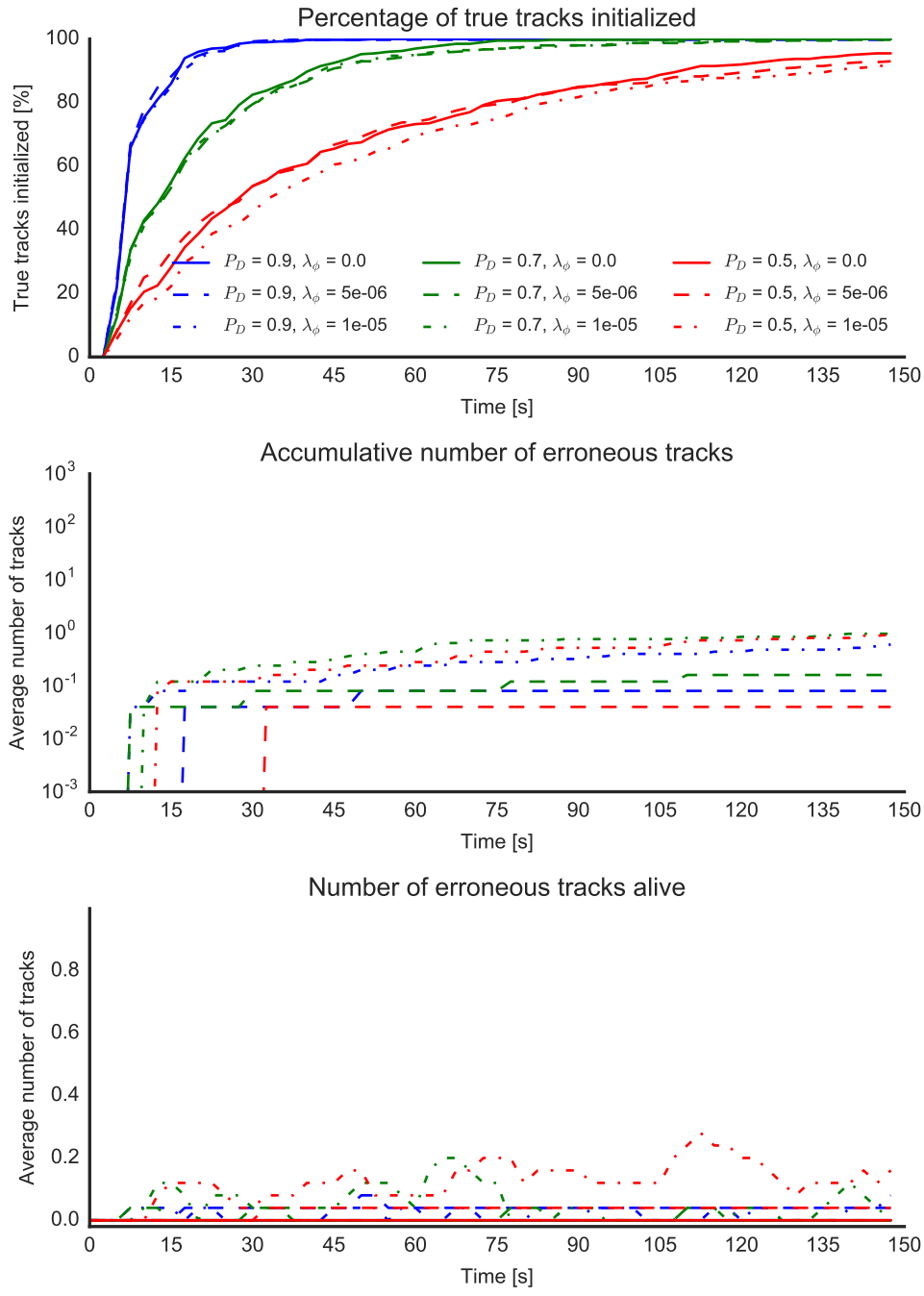

Figure B.29: Scenario 2 — Initialization time (2/2)

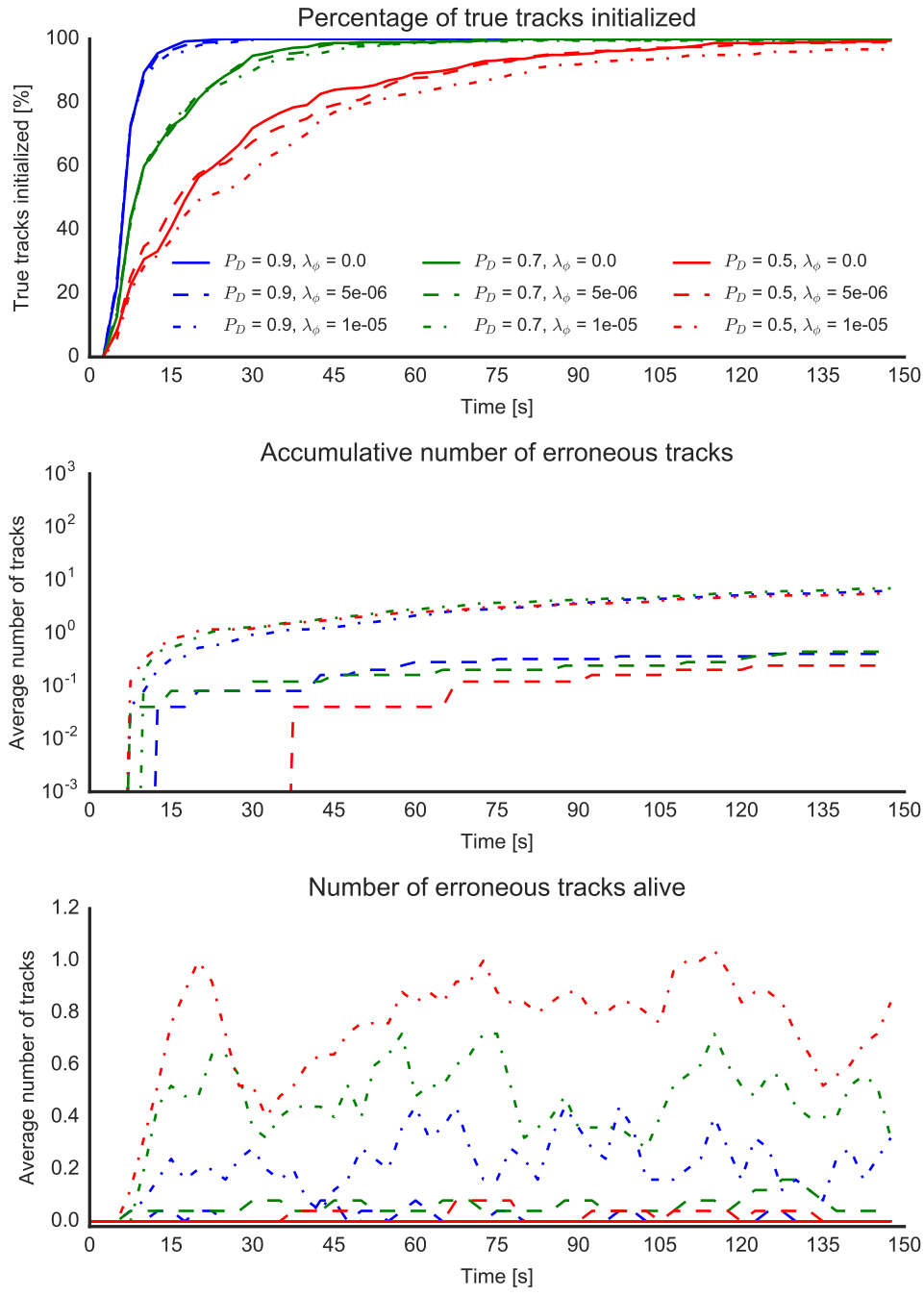

Figure B.30: Scenario 2 — Initialization time (2/3)

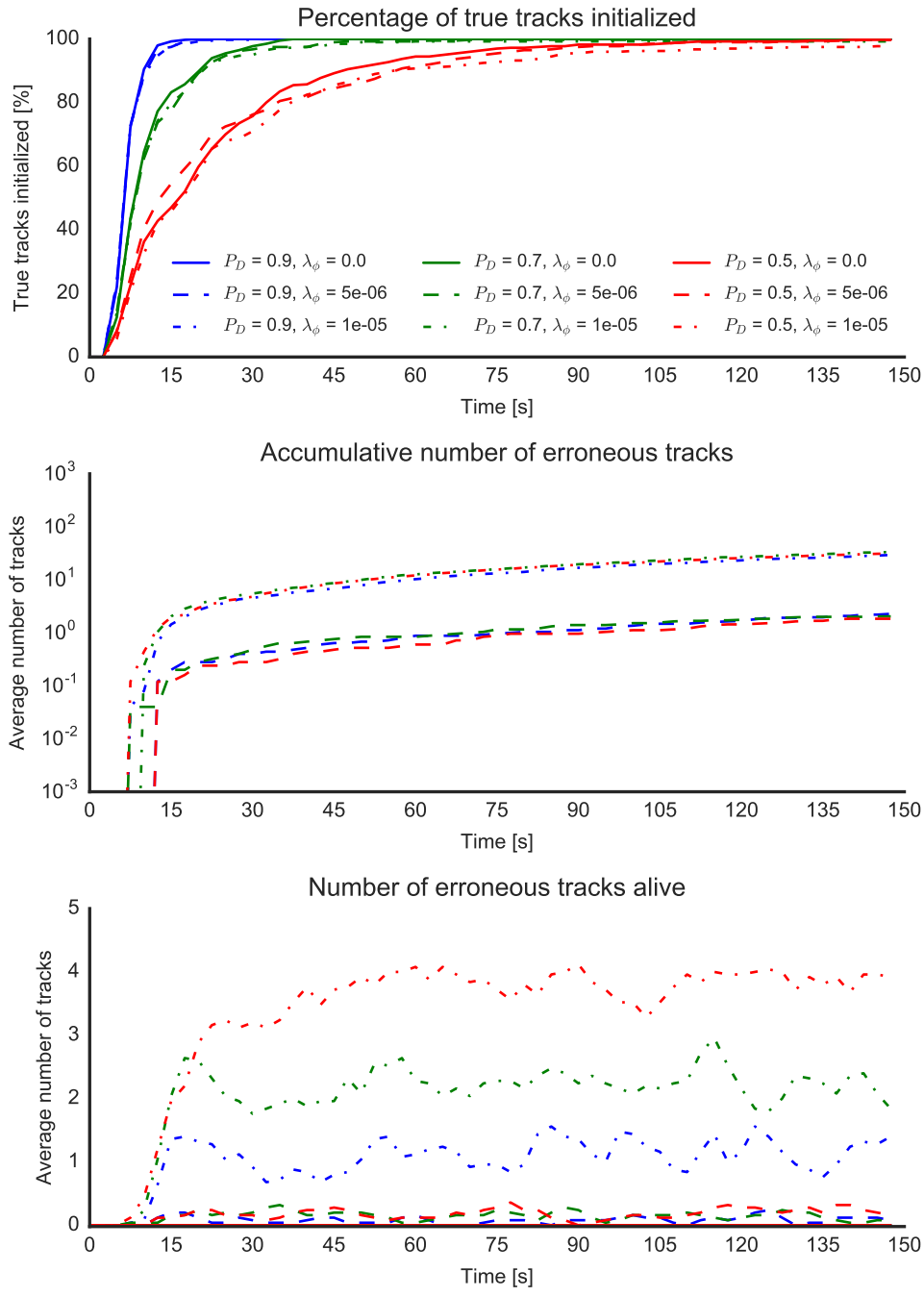

Figure B.31: Scenario 2 — Initialization time (2/4)

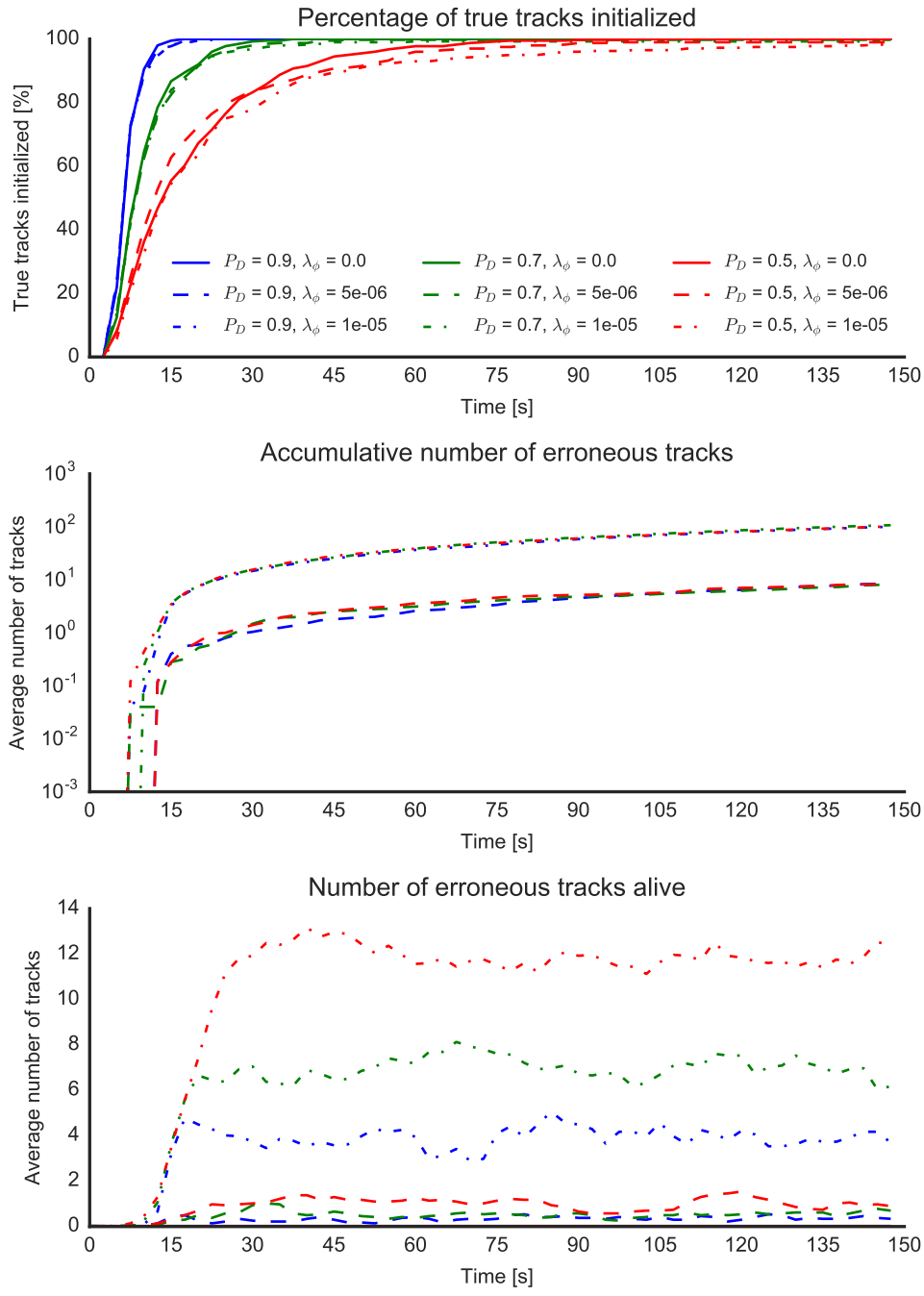

Figure B.32: Scenario 2 — Initialization time (2/5)

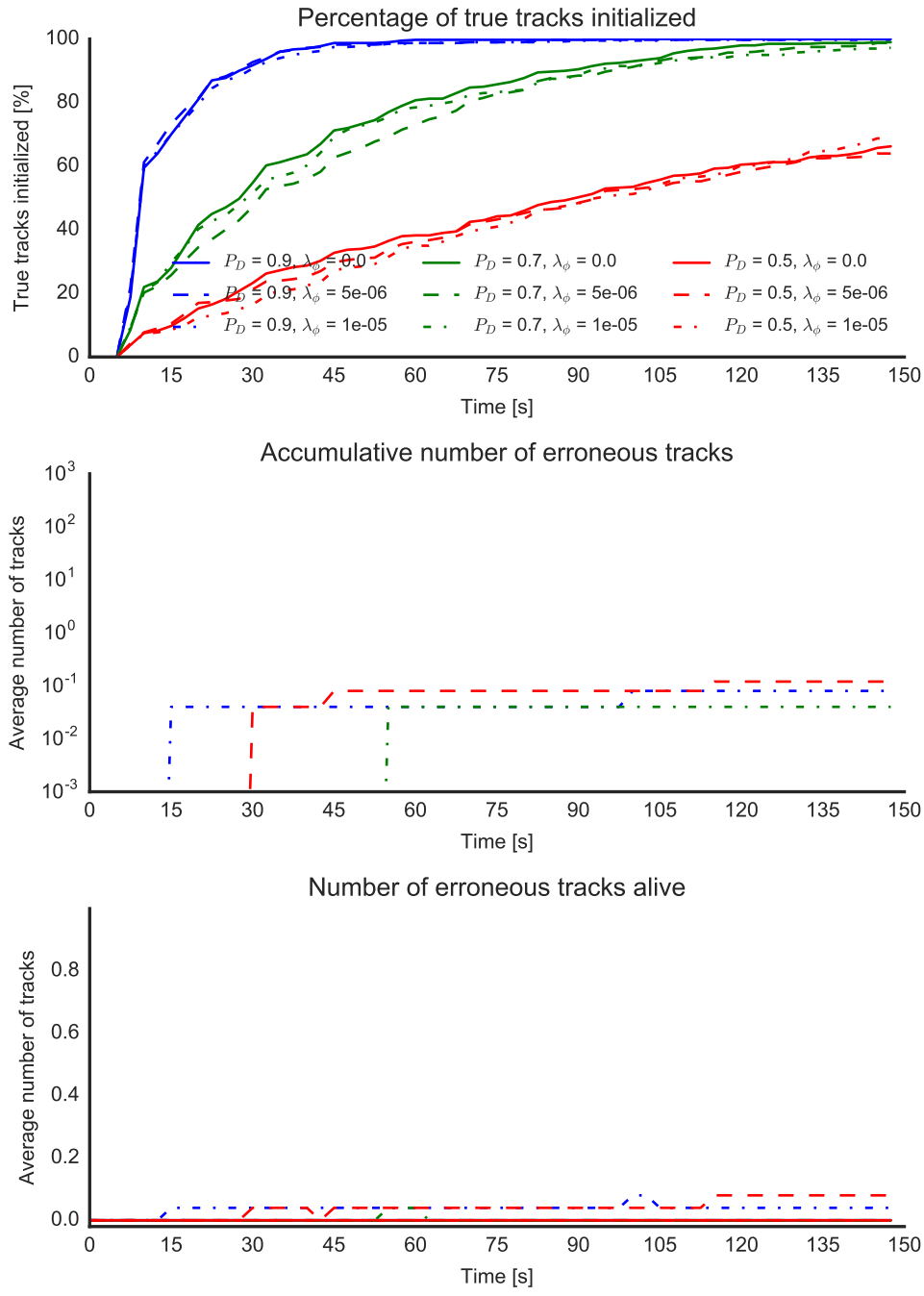

Figure B.33: Scenario 2 — Initialization time (3/3)

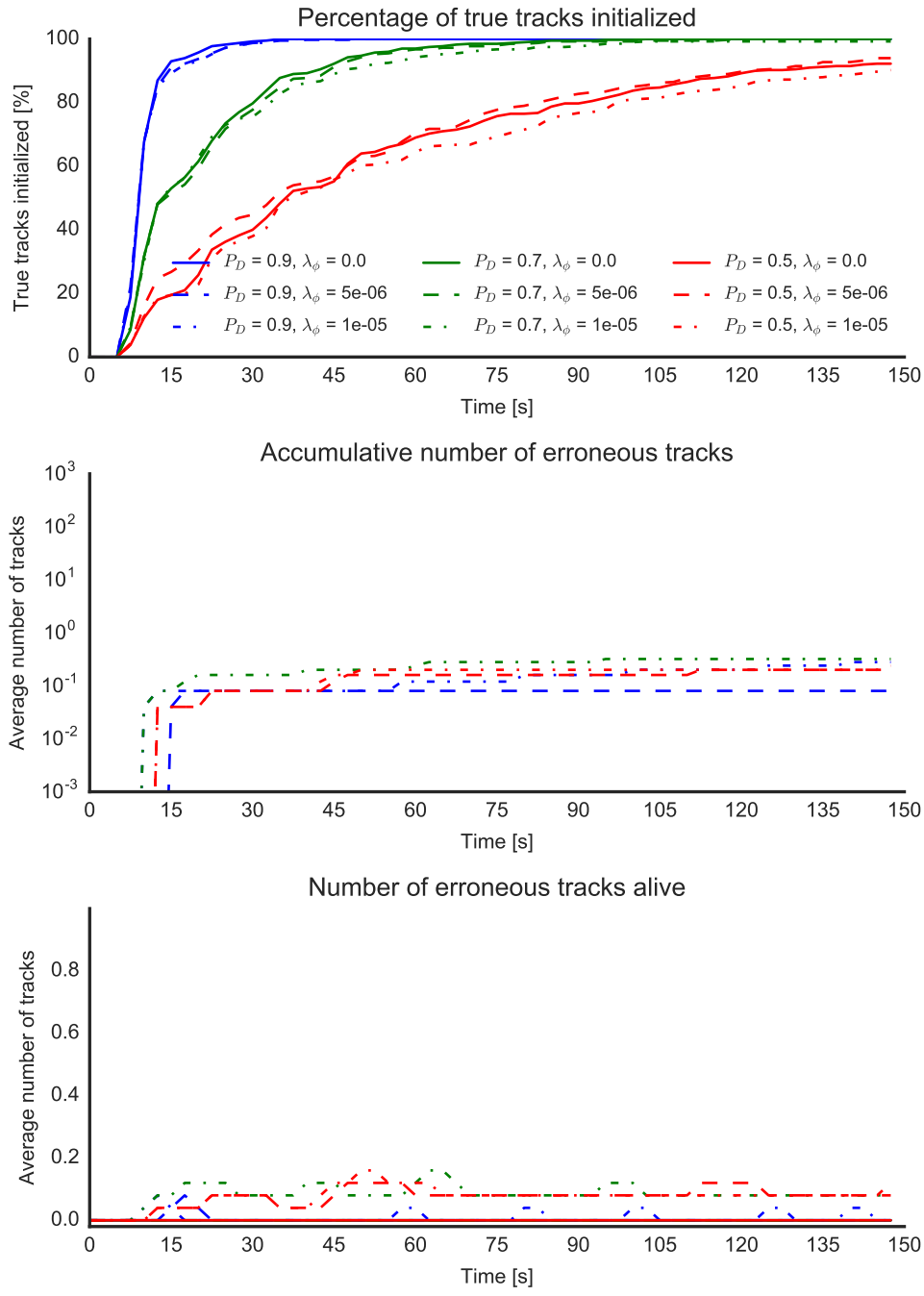

Figure B.34: Scenario 2 — Initialization time (3/4)

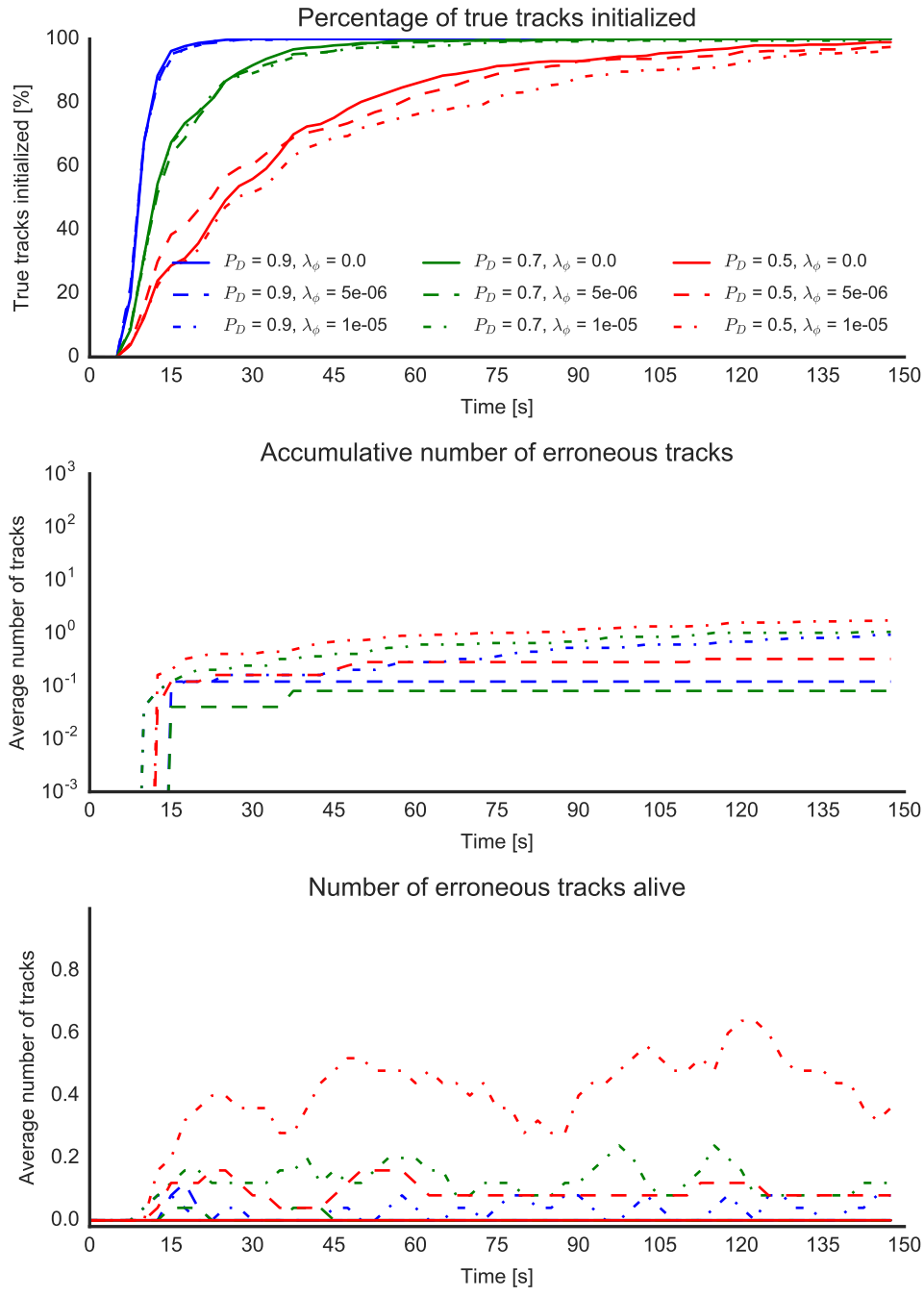

Figure B.35: Scenario 2 — Initialization time (3/5)

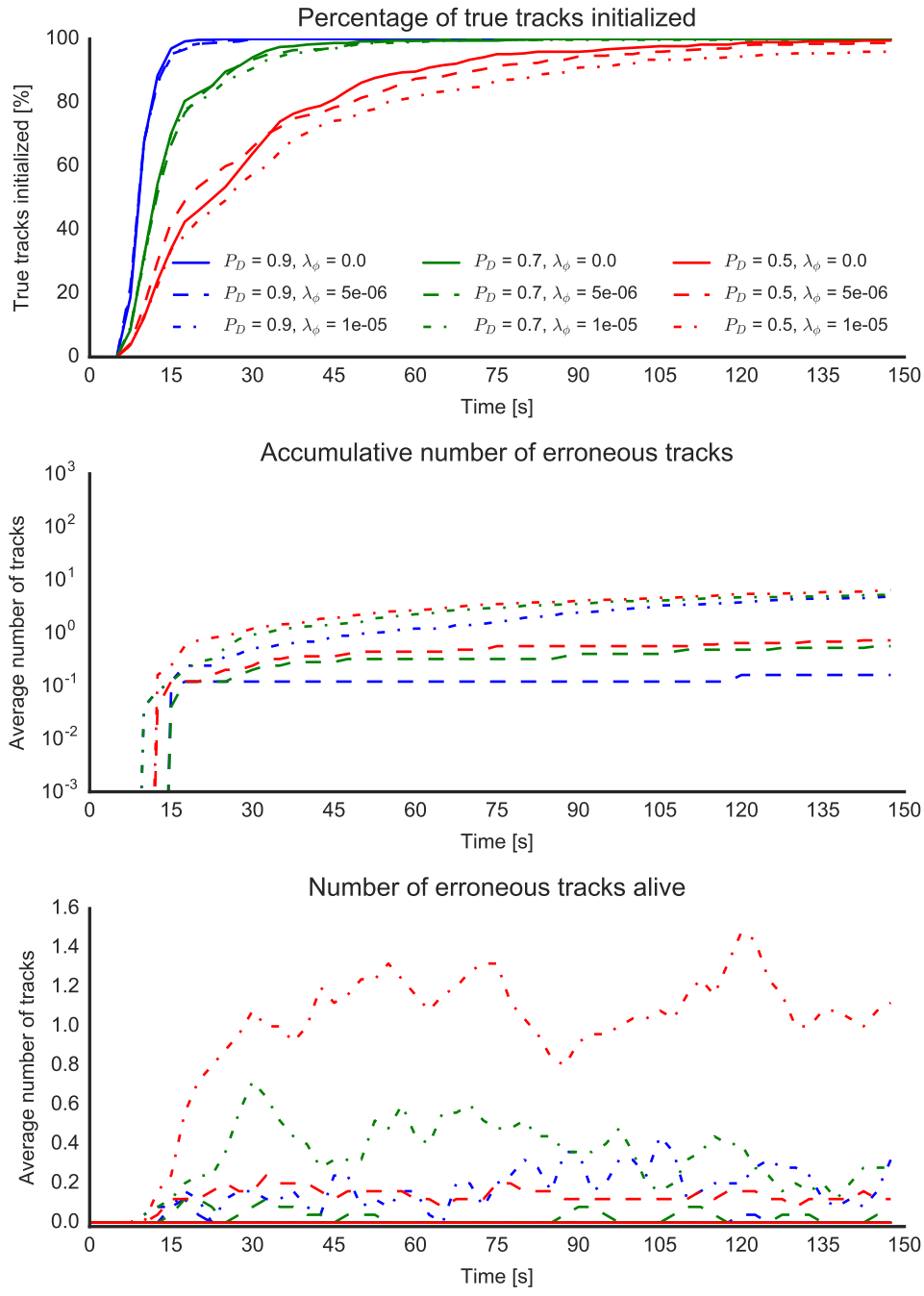

Figure B.36: Scenario 2 — Initialization time (3/6)

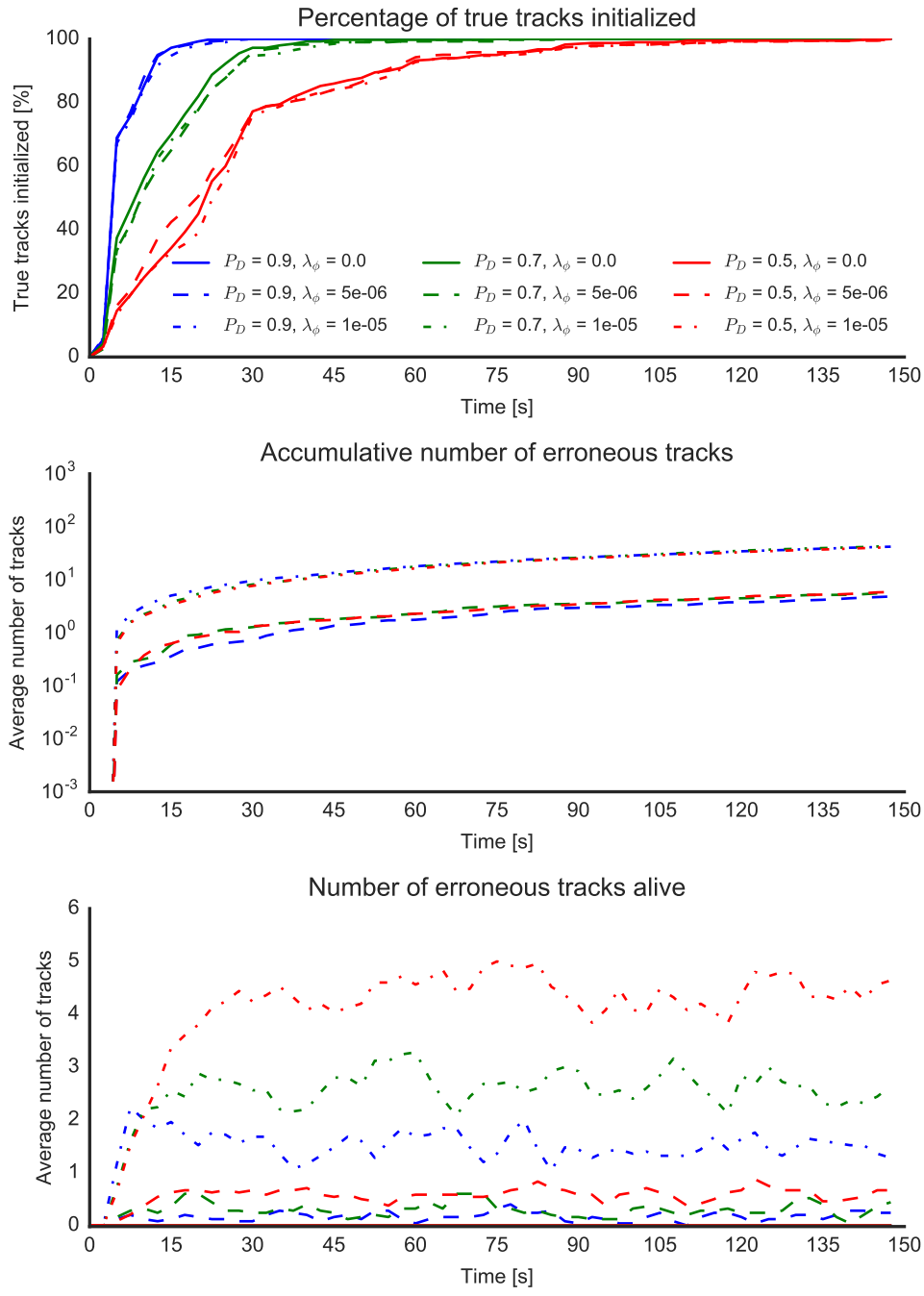

Figure B.37: Scenario 3 — Initialization time (1/1)

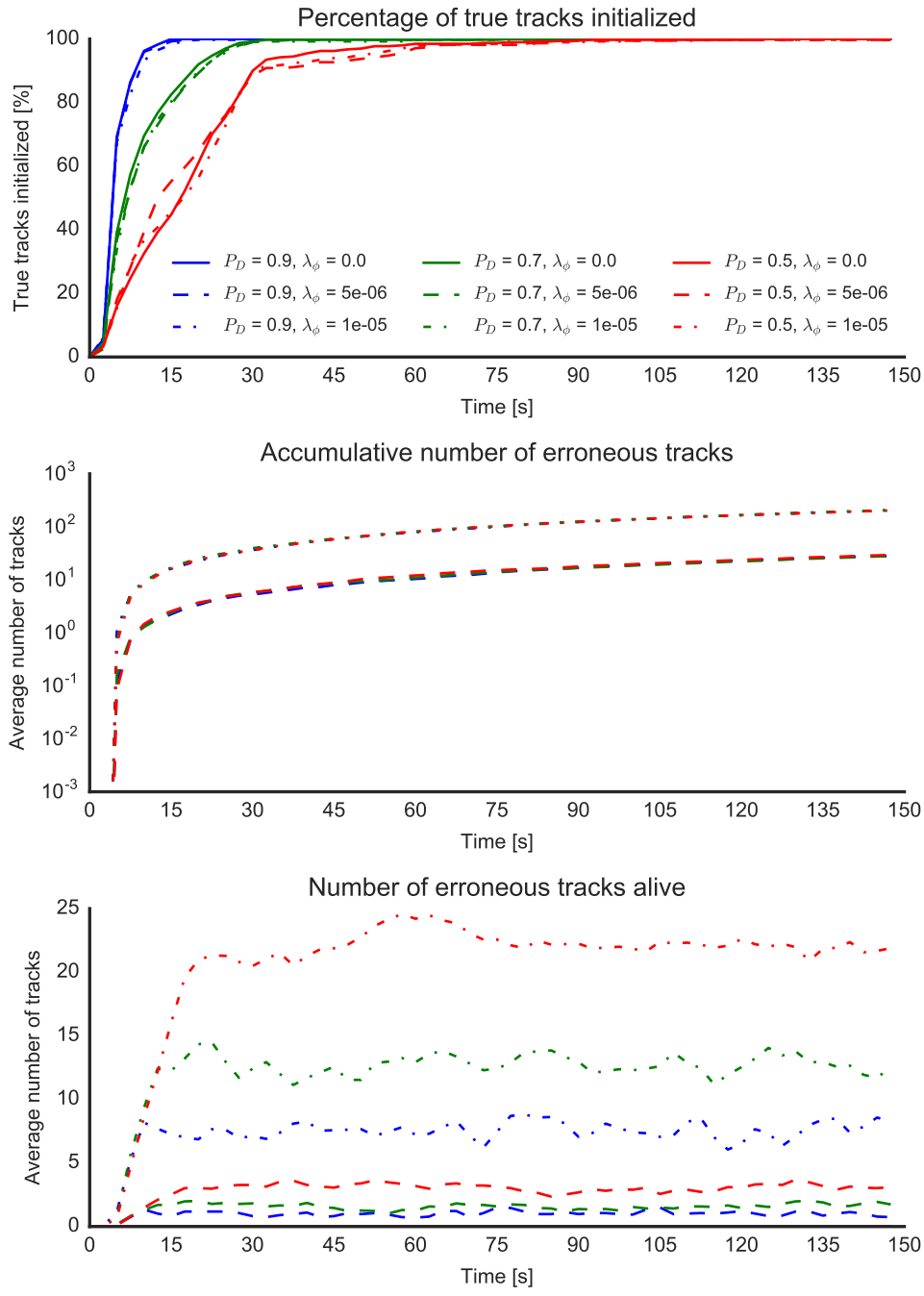

Figure B.38: Scenario 3 — Initialization time (1/2)

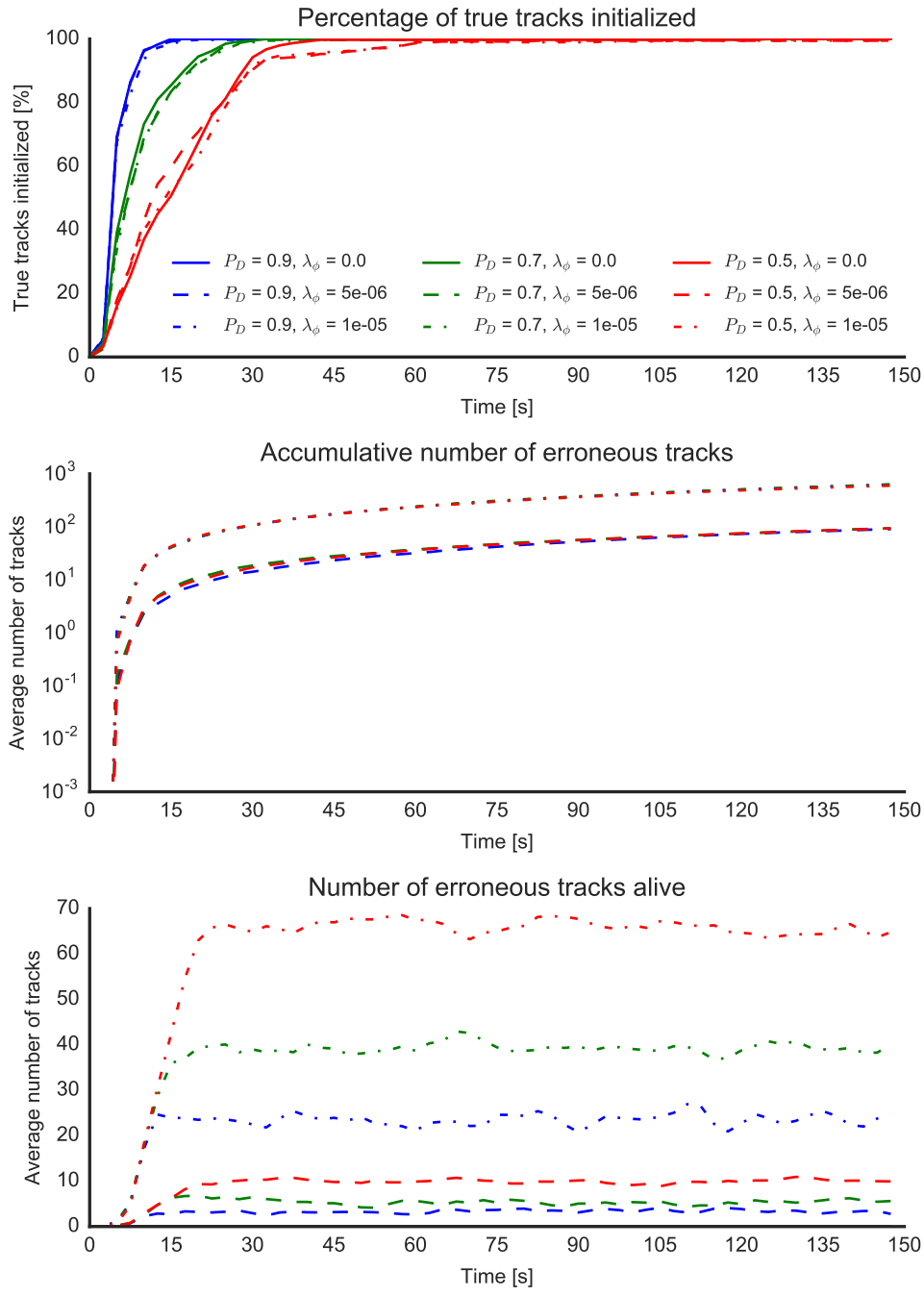

Figure B.39: Scenario 3 — Initialization time (1/3)

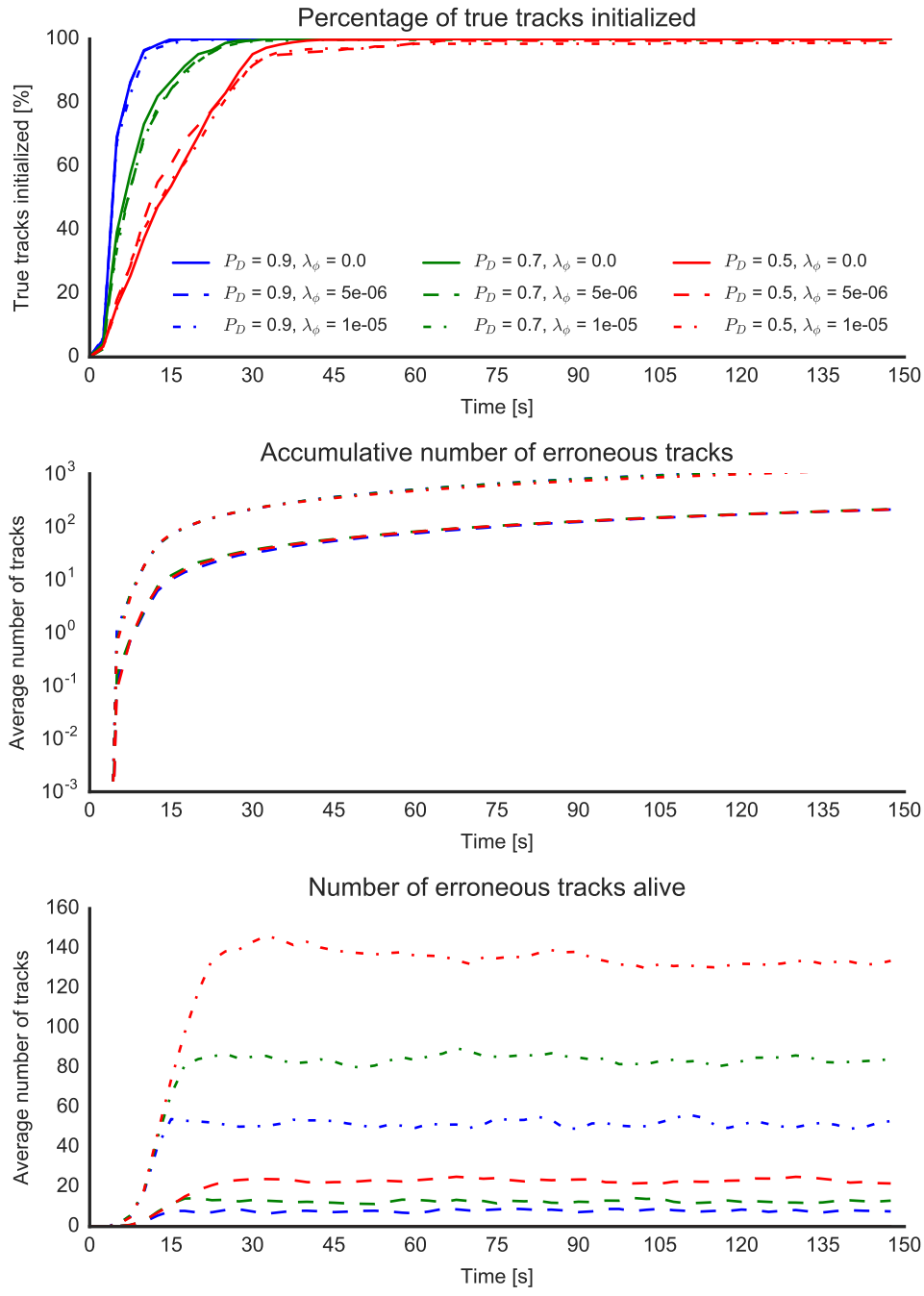

Figure B.40: Scenario 3 — Initialization time (1/4)

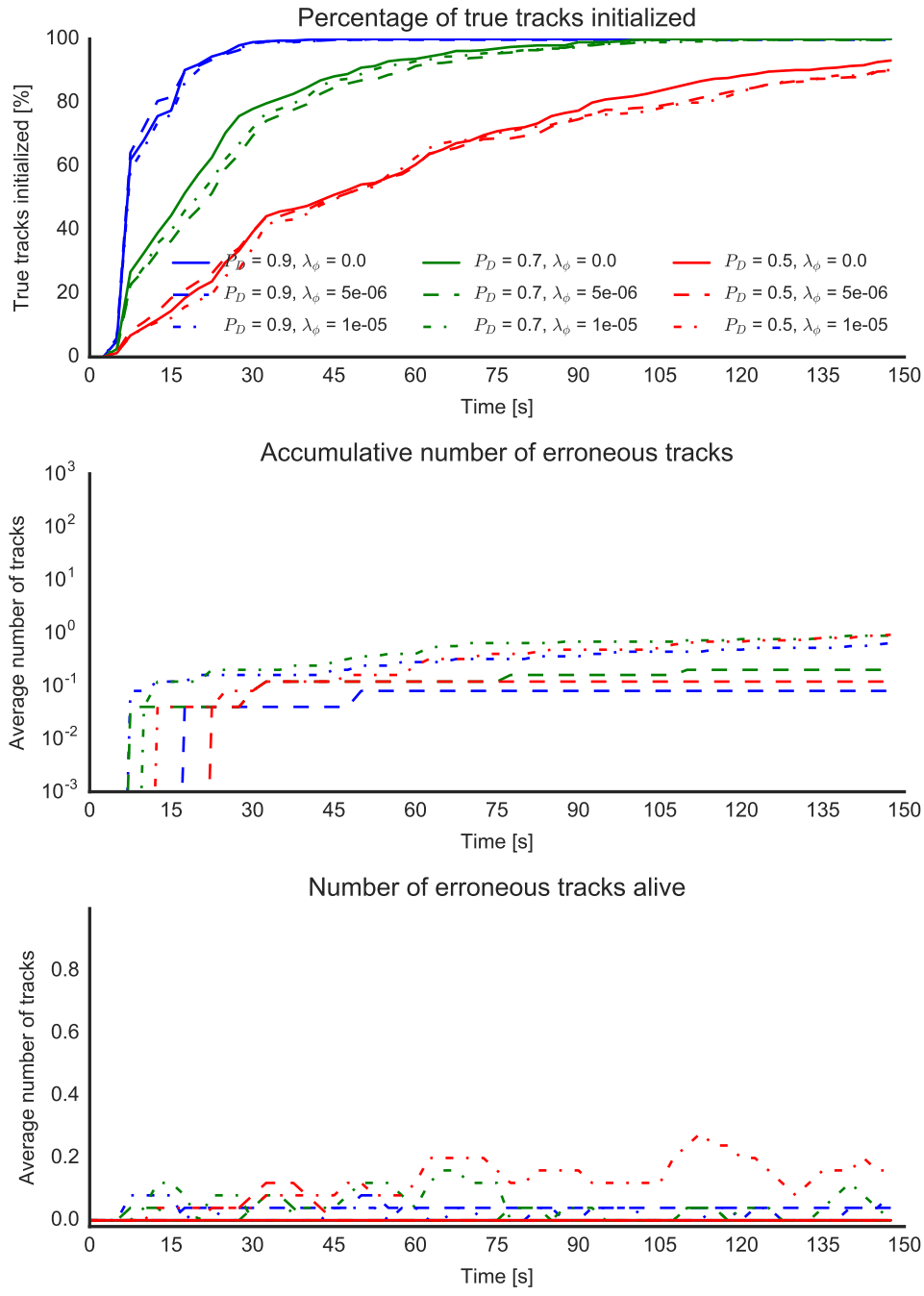

Figure B.41: Scenario 3 — Initialization time (2/2)

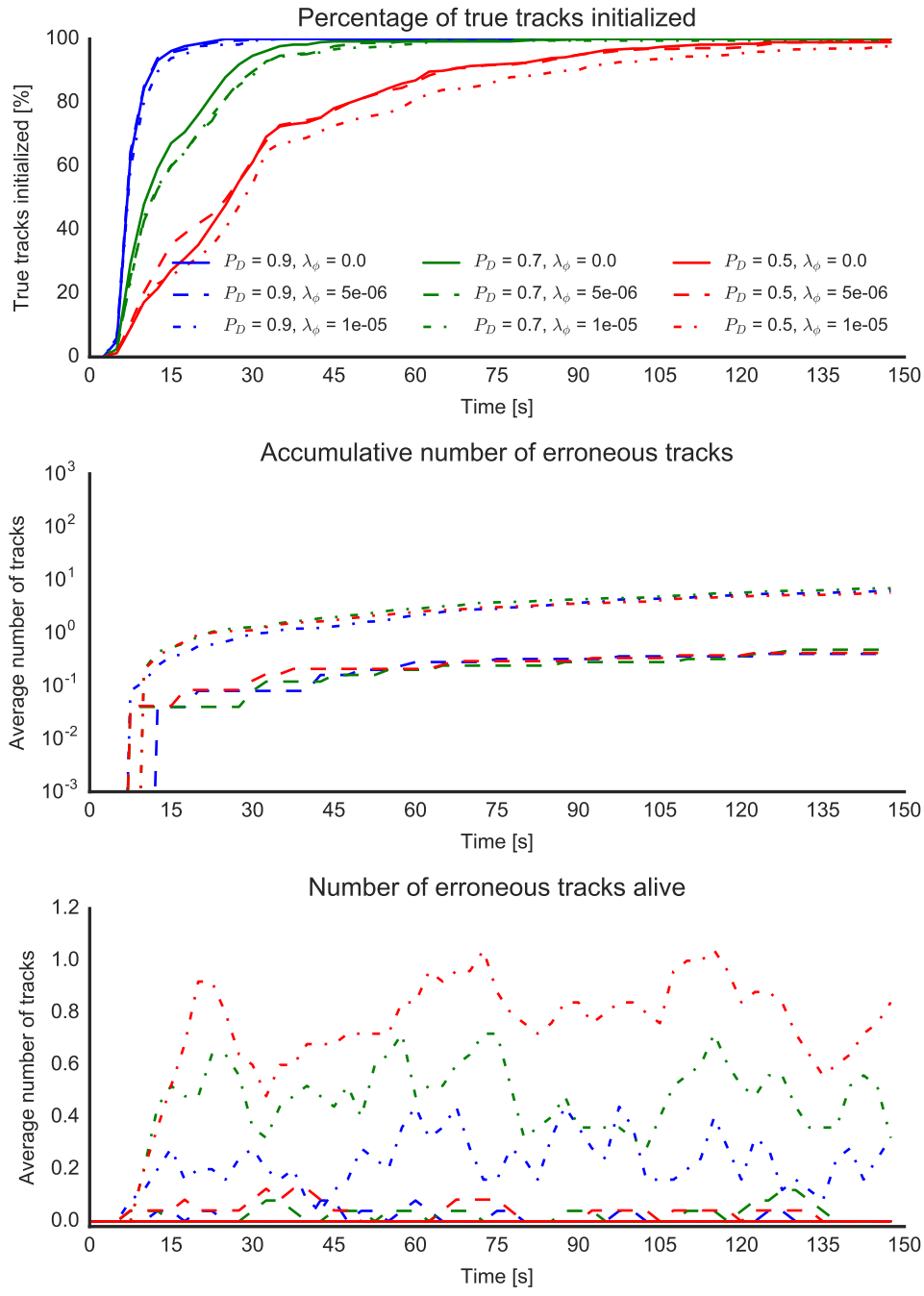

Figure B.42: Scenario 3 — Initialization time (2/3)

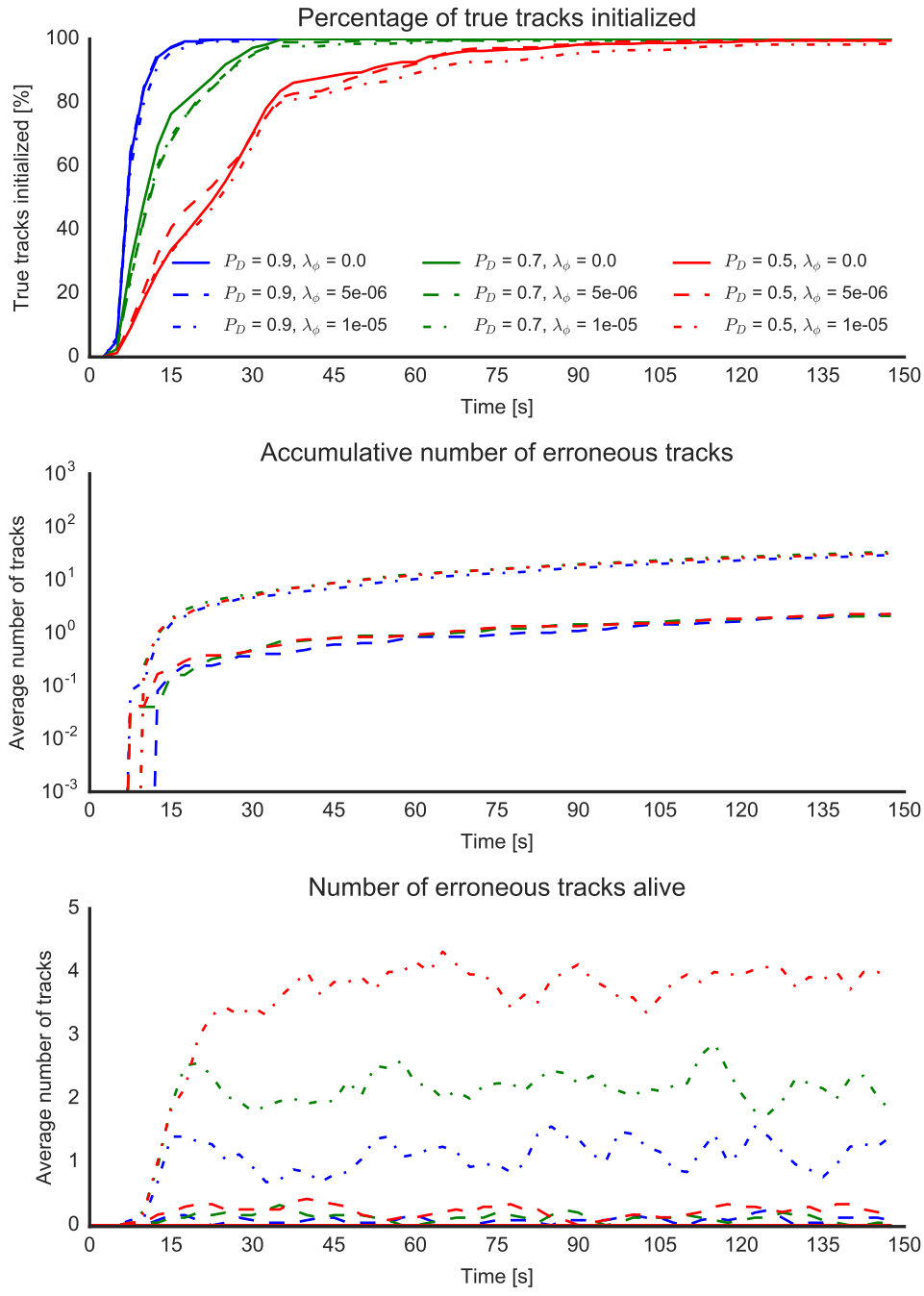

Figure B.43: Scenario 3 — Initialization time (2/4)

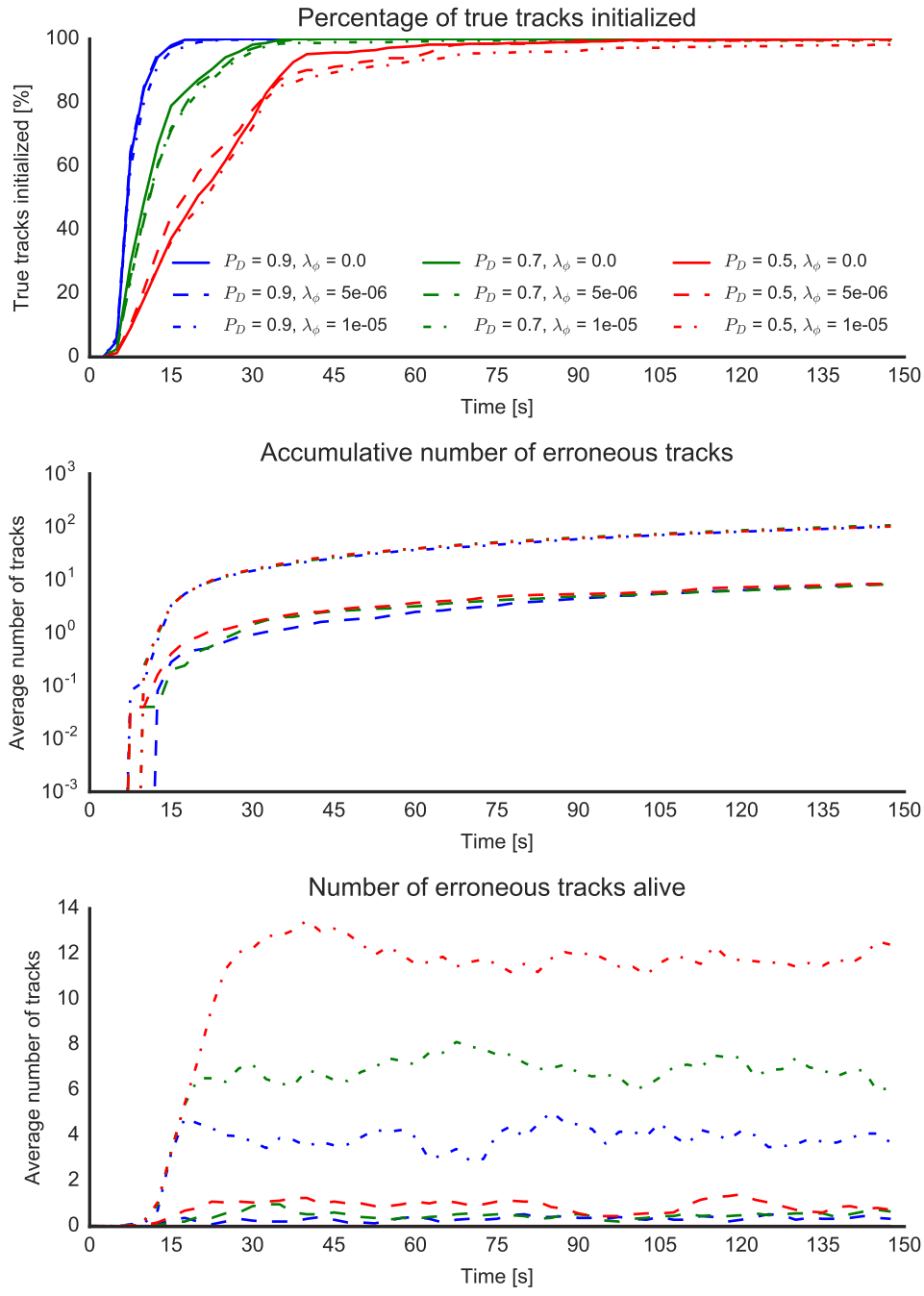

Figure B.44: Scenario 3 — Initialization time (2/5)

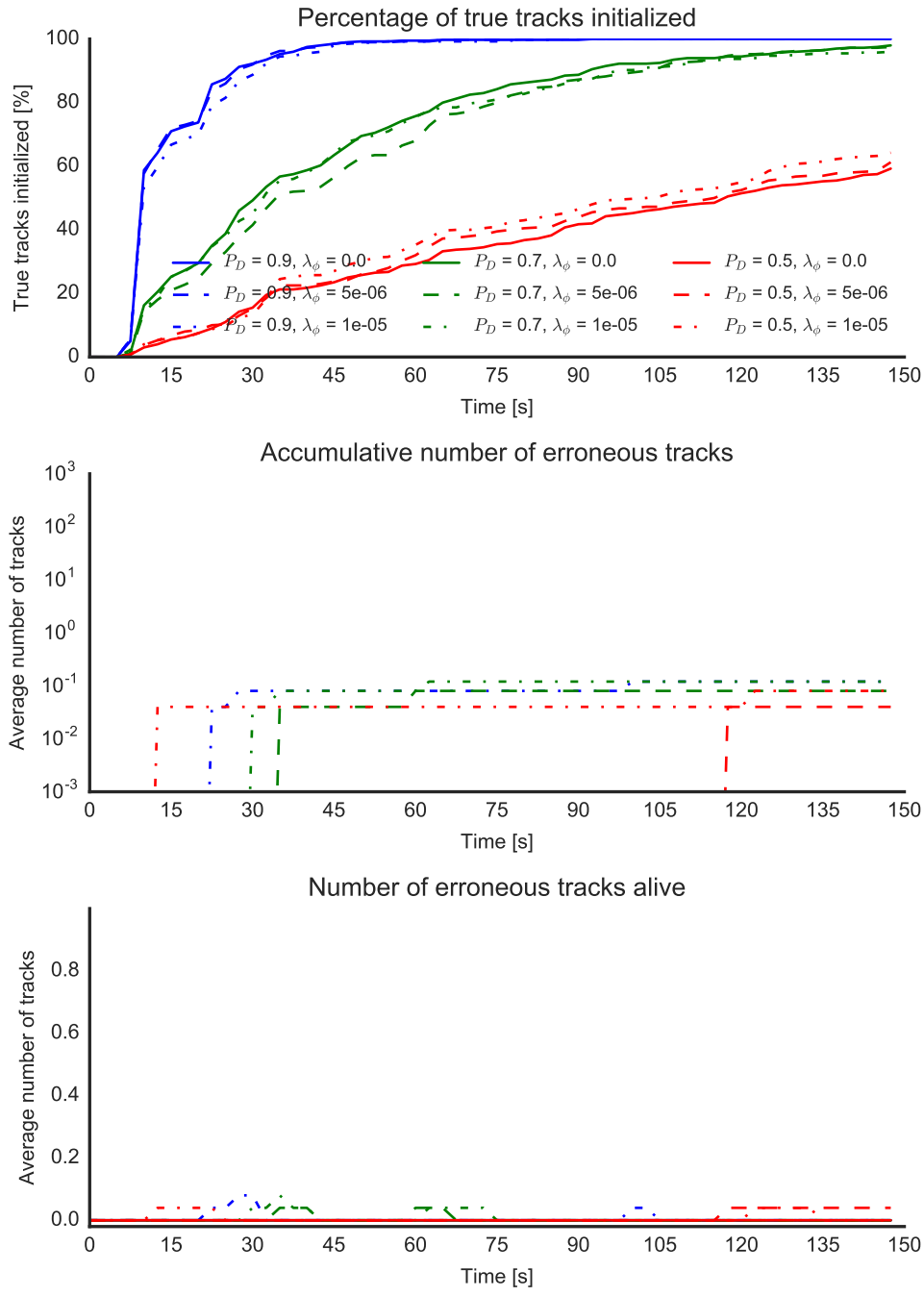

Figure B.45: Scenario 3 — Initialization time (3/3)

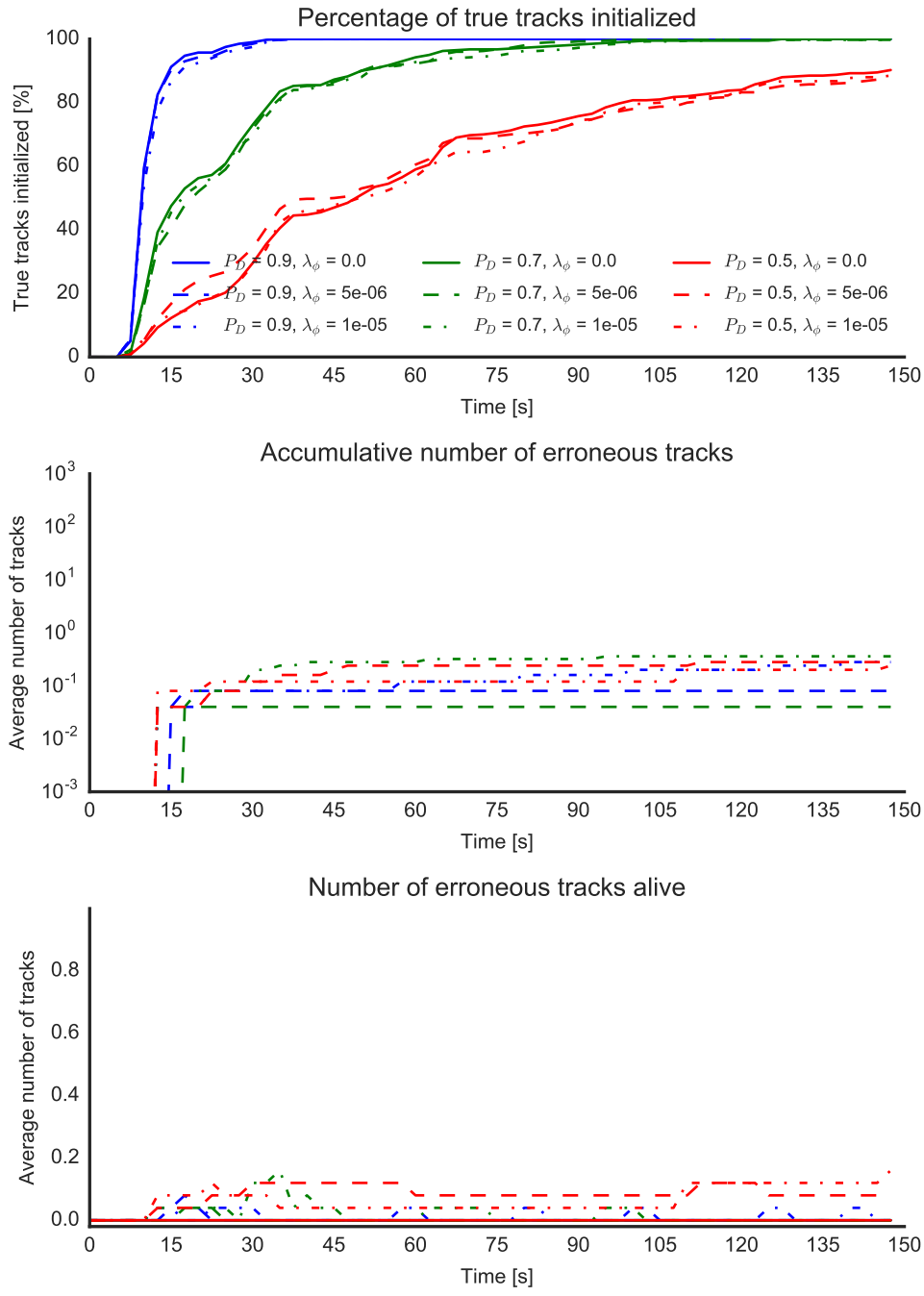

Figure B.46: Scenario 3 — Initialization time (3/4)

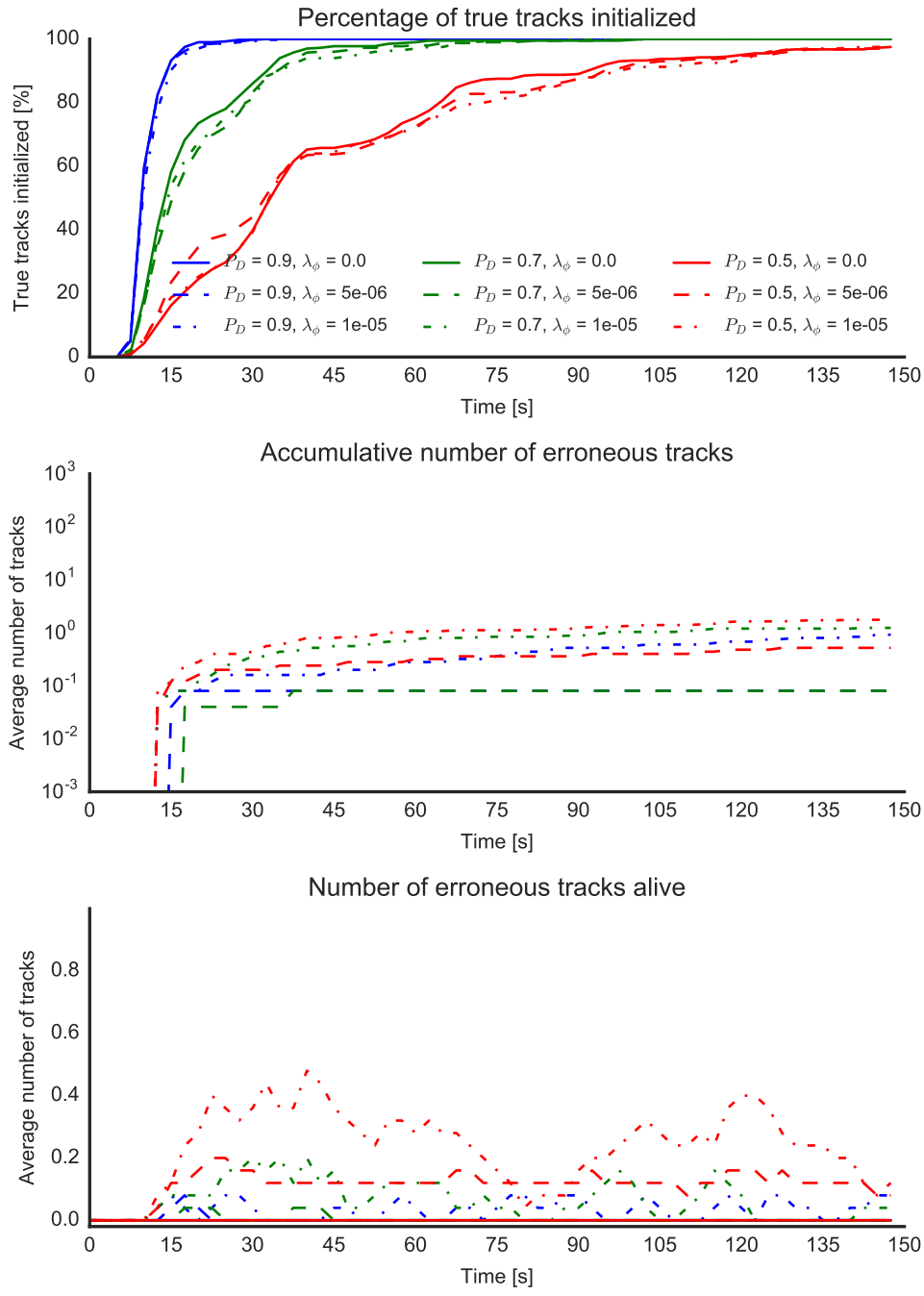

Figure B.47: Scenario 3 — Initialization time (3/5)

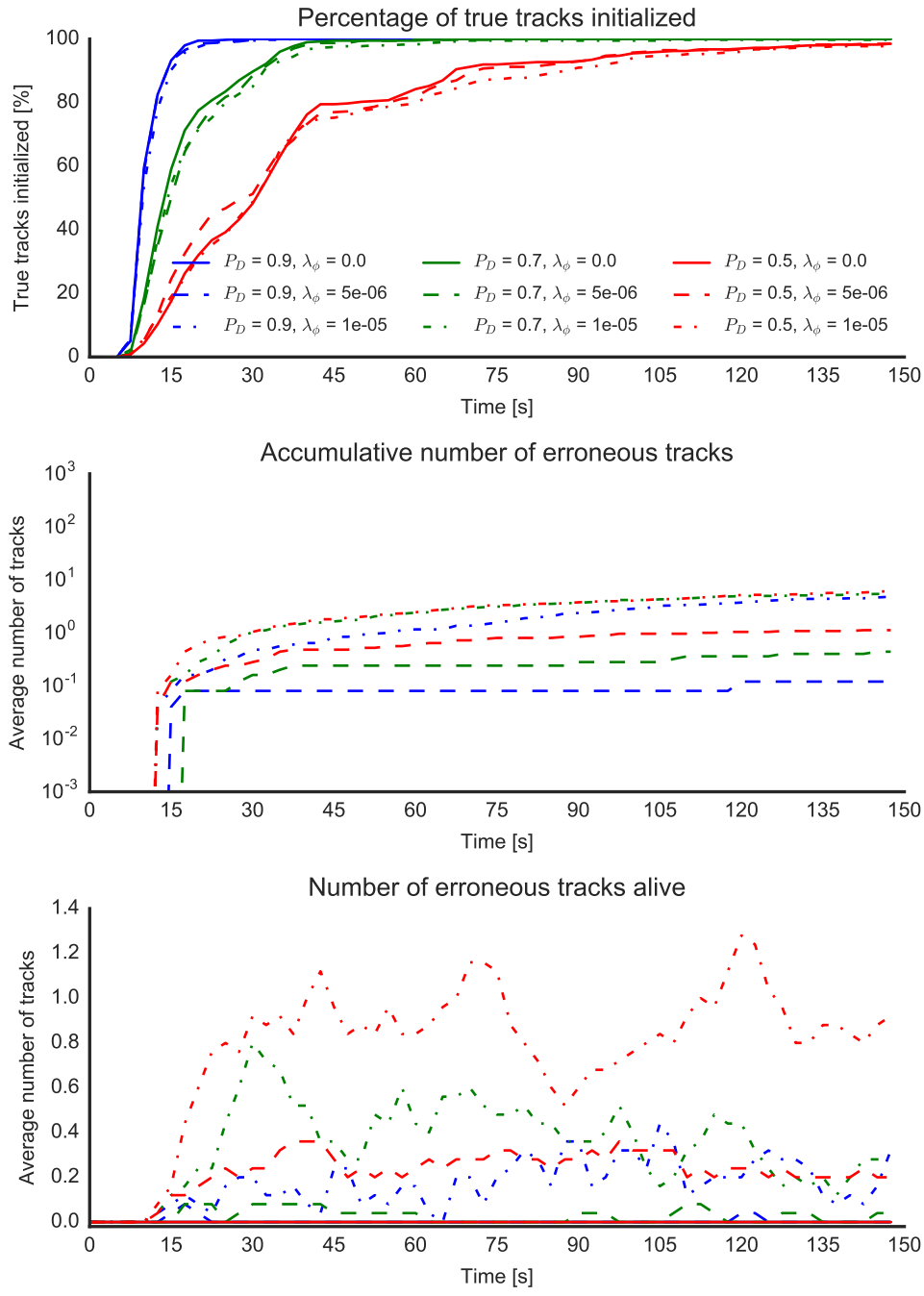

Figure B.48: Scenario 3 — Initialization time (3/6)

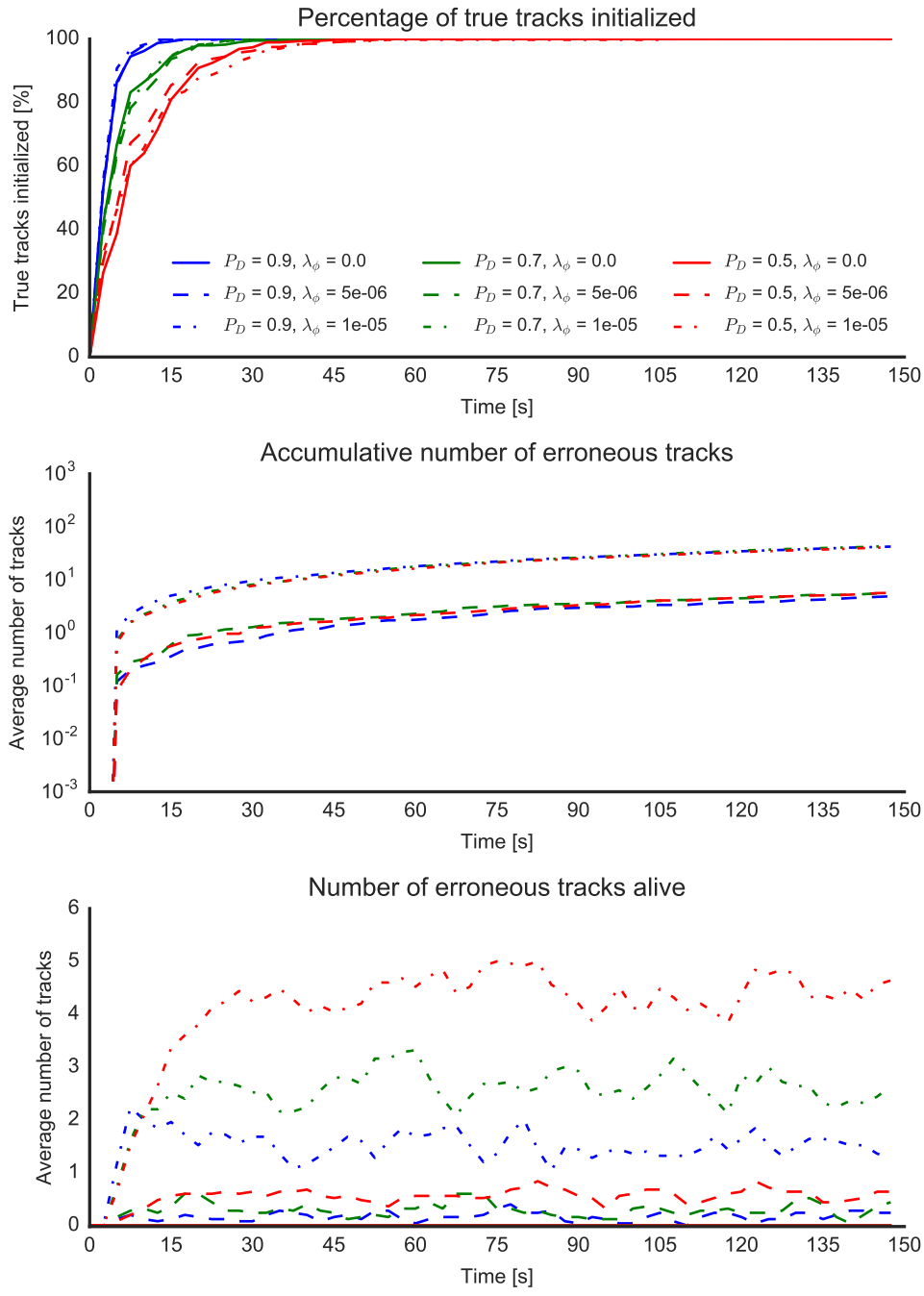

Figure B.49: Scenario 4 — Initialization time (1/1)

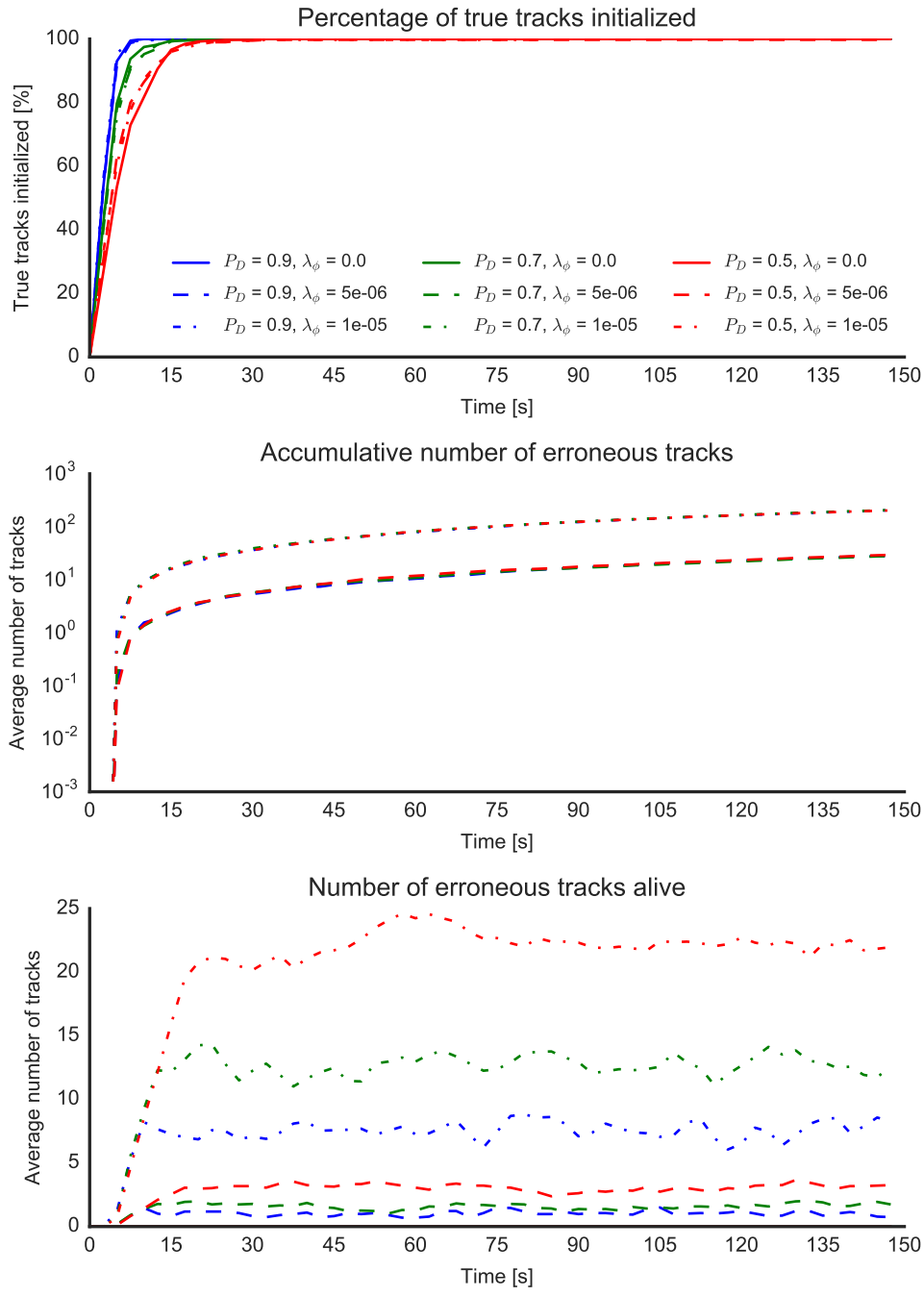

Figure B.50: Scenario 4 — Initialization time (1/2)

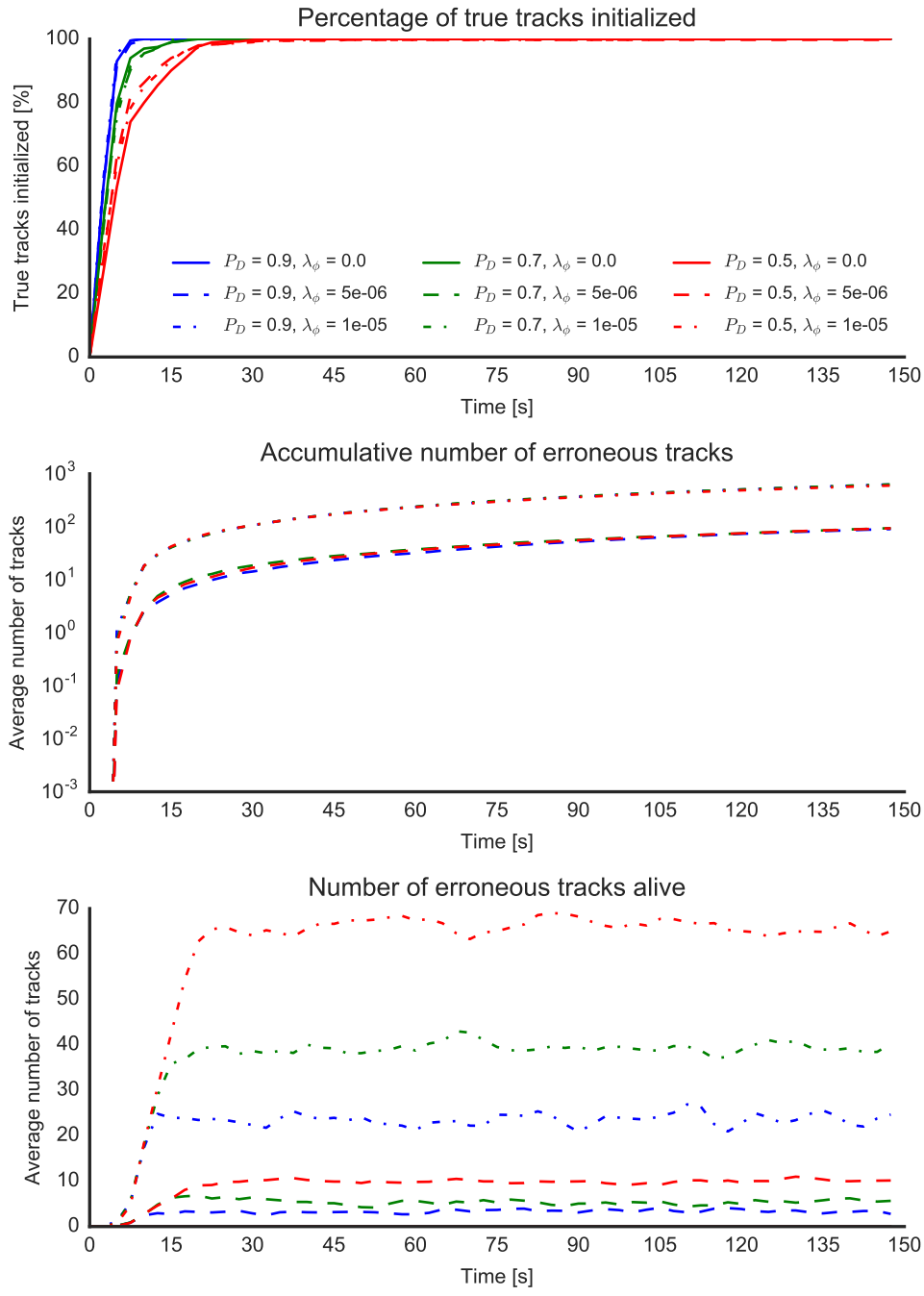

Figure B.51: Scenario 4 — Initialization time (1/3)

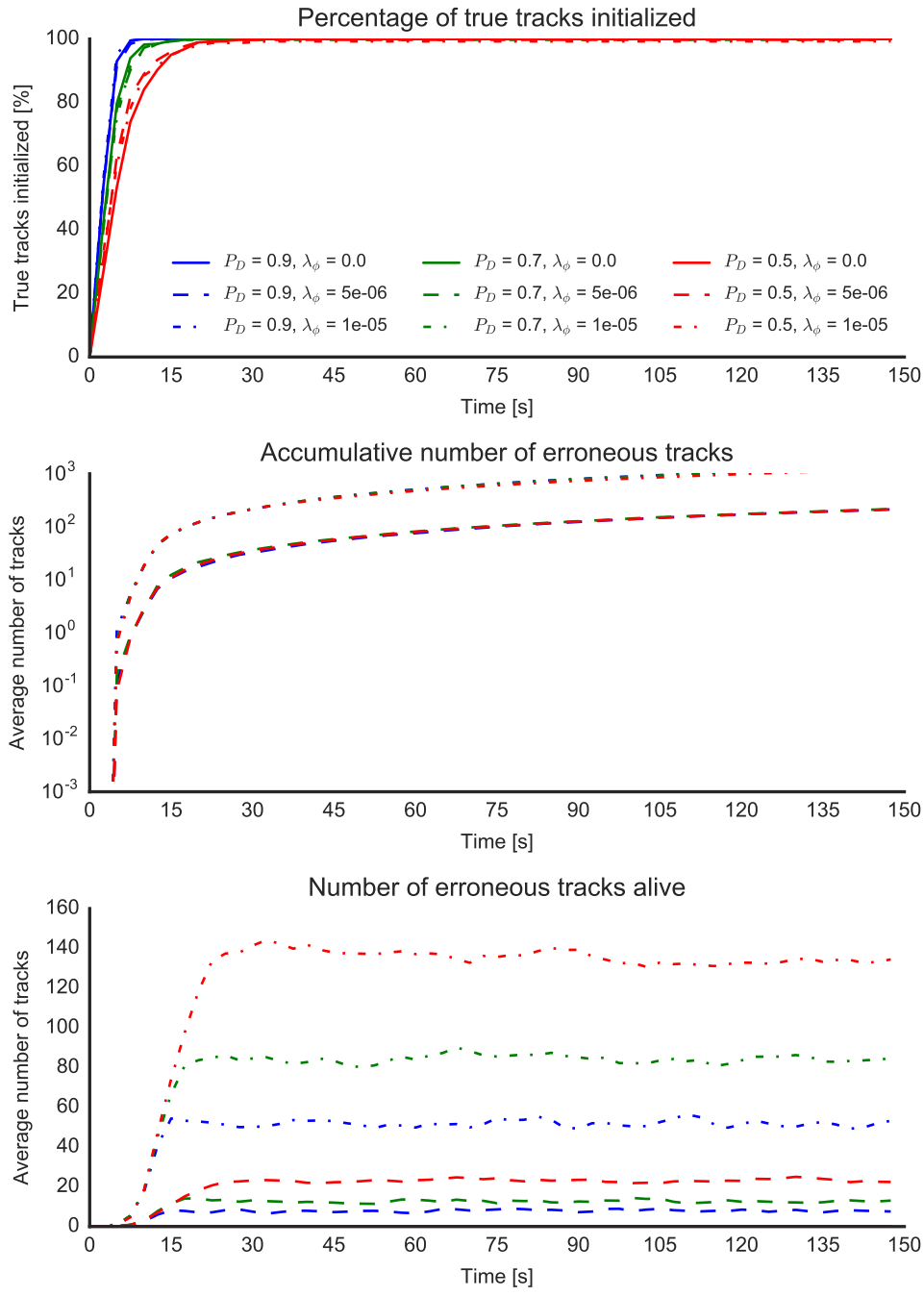

Figure B.52: Scenario 4 — Initialization time (1/4)

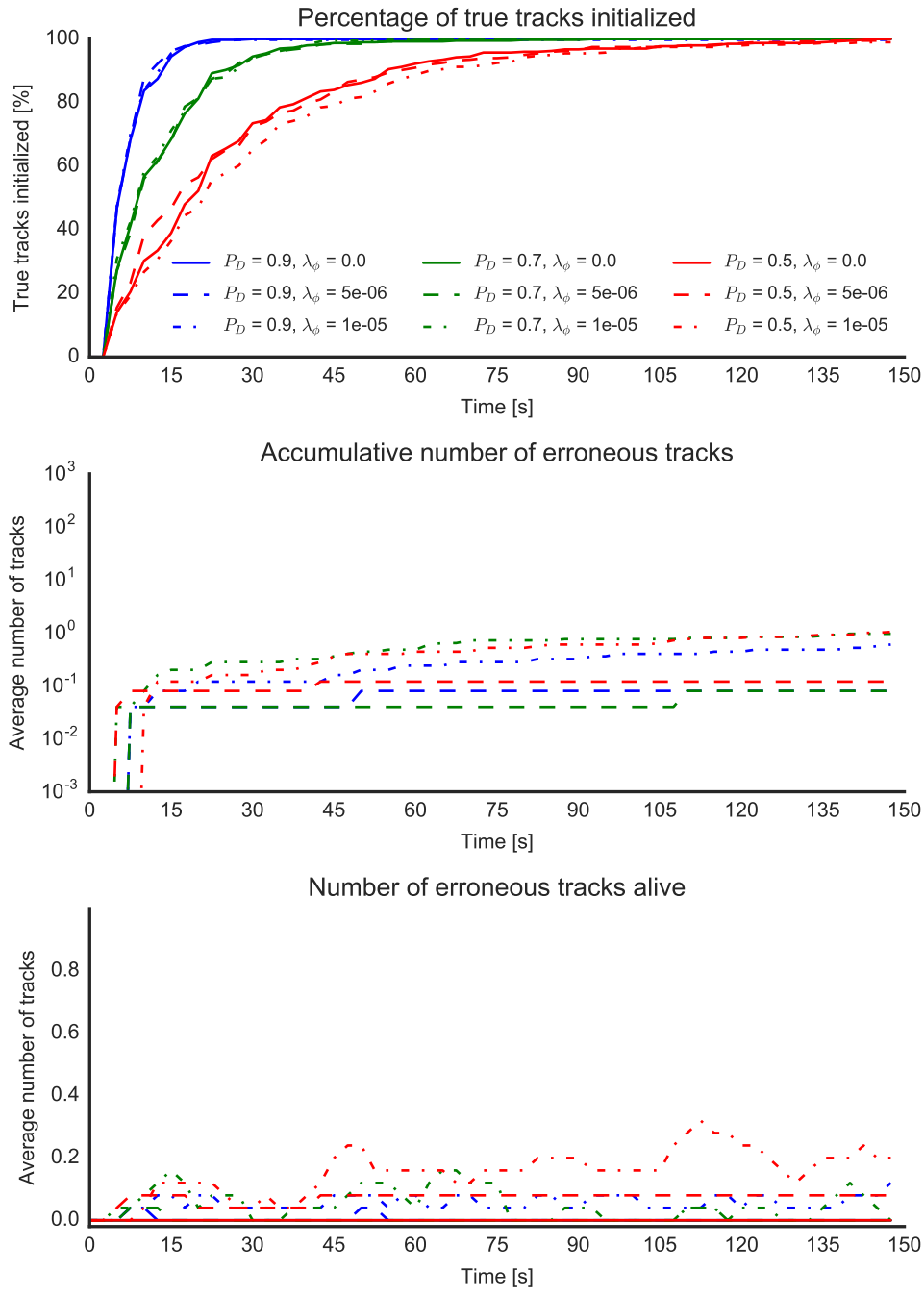

Figure B.53: Scenario 4 — Initialization time (2/2)

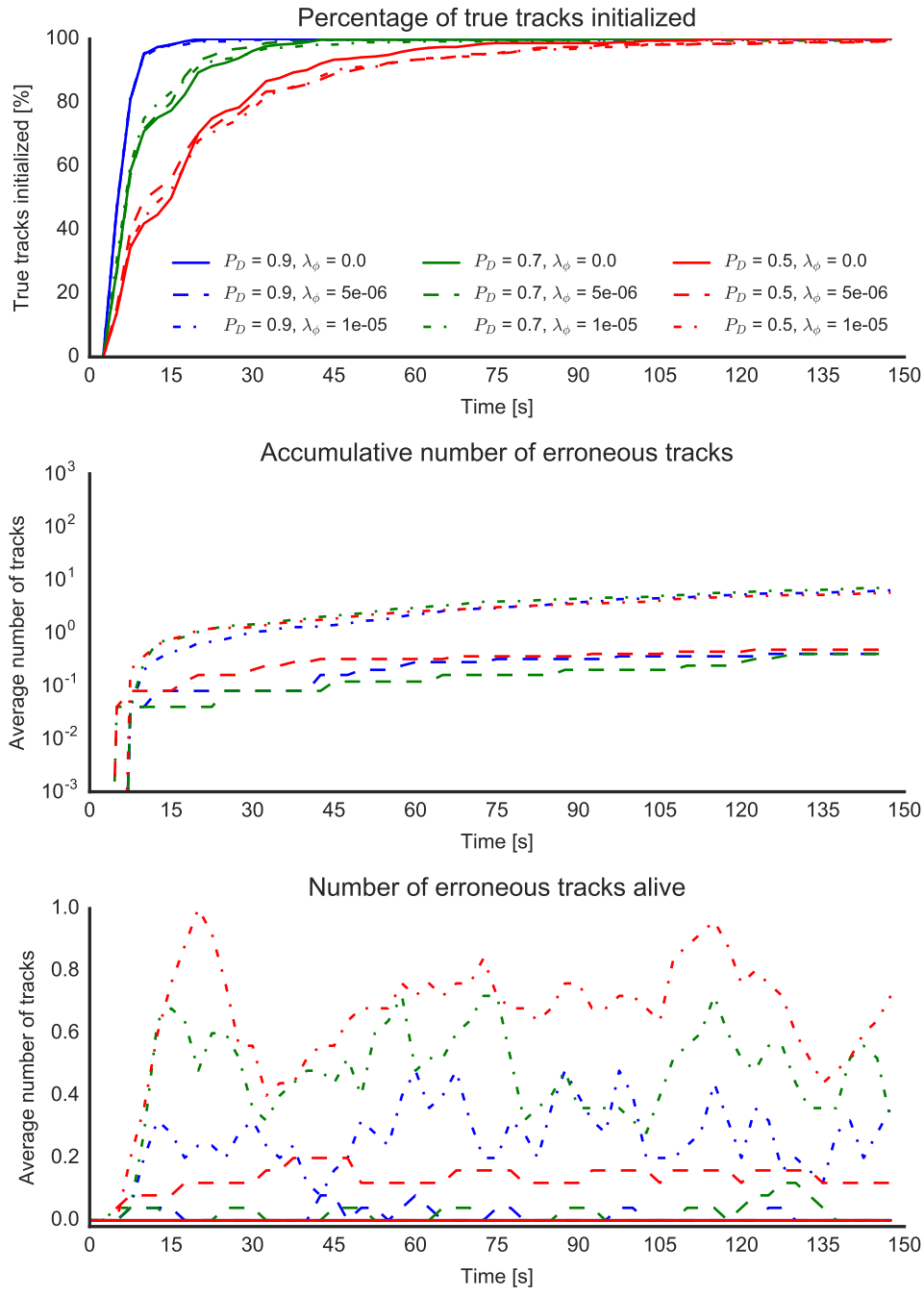

Figure B.54: Scenario 4 — Initialization time (2/3)

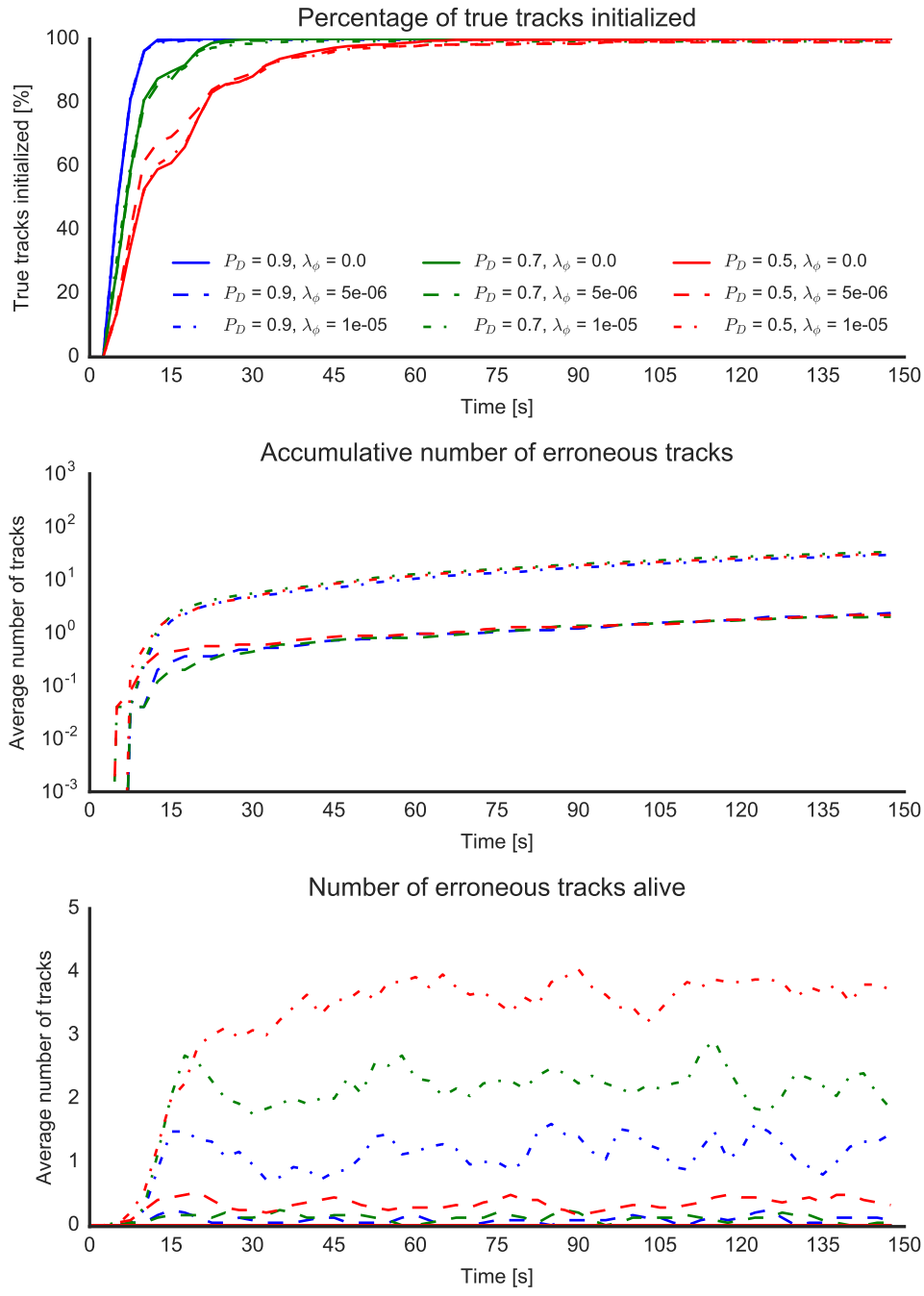

Figure B.55: Scenario 4 — Initialization time (2/4)

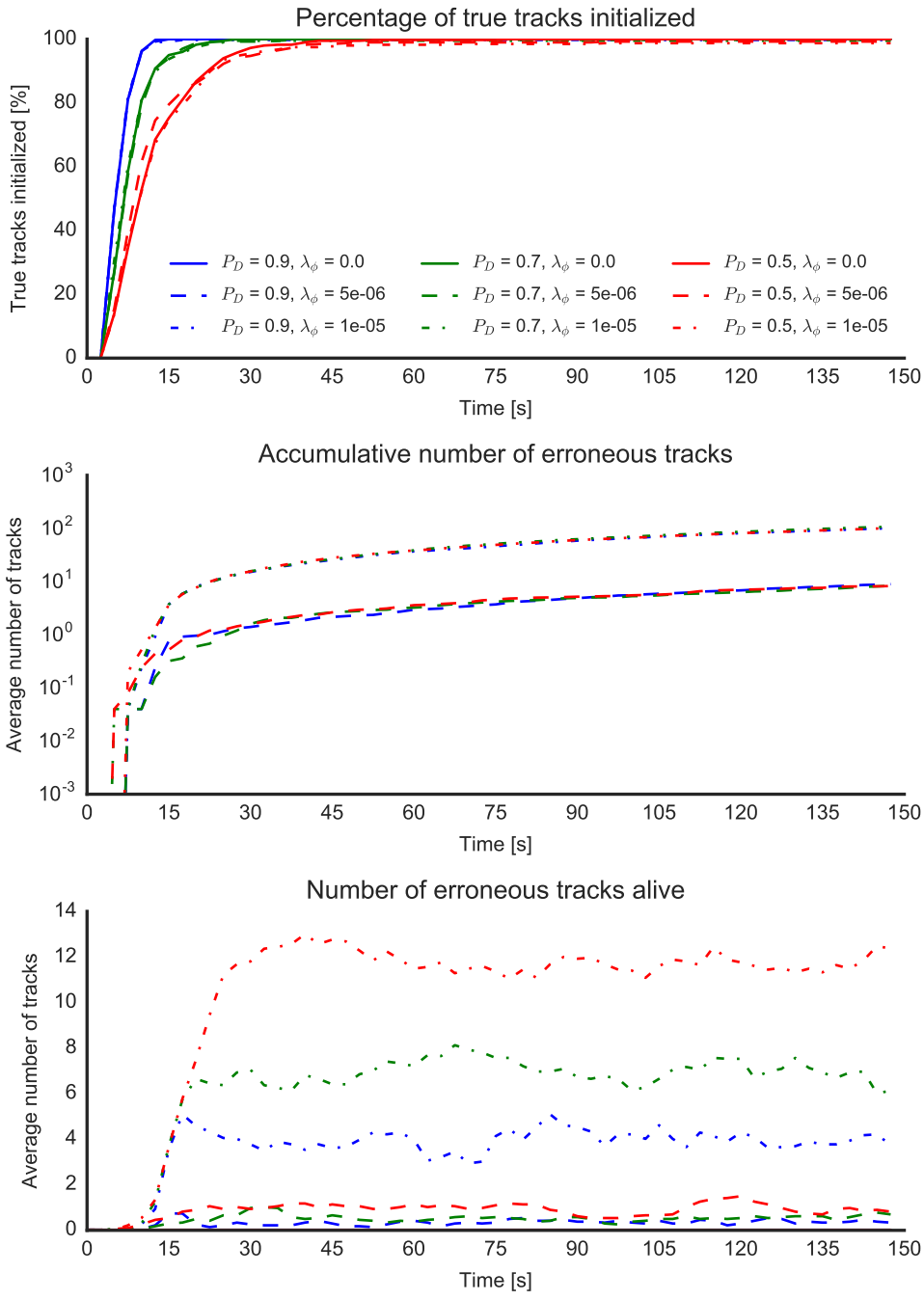

Figure B.56: Scenario 4 — Initialization time (2/5)

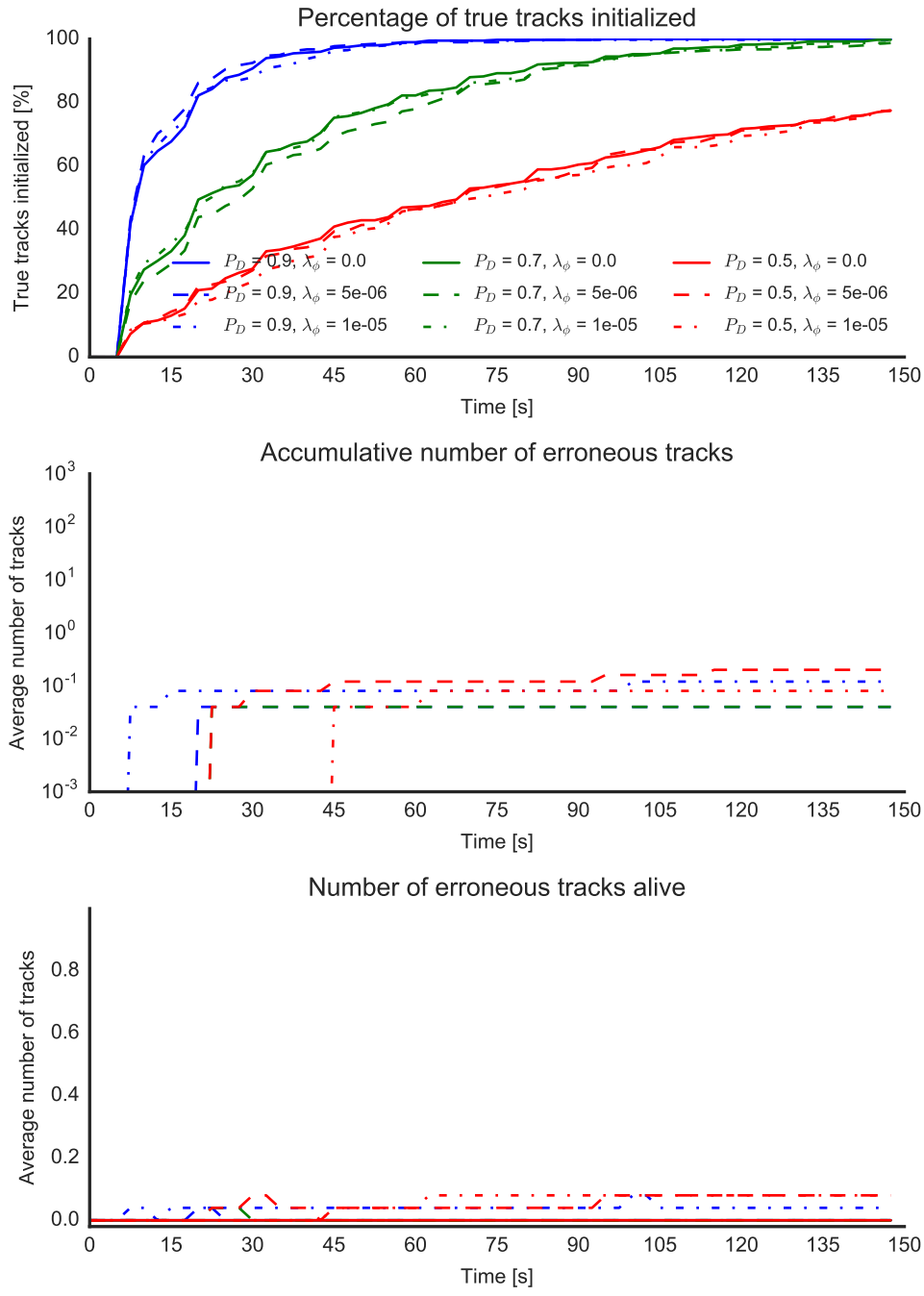

Figure B.57: Scenario 4 — Initialization time (3/3)

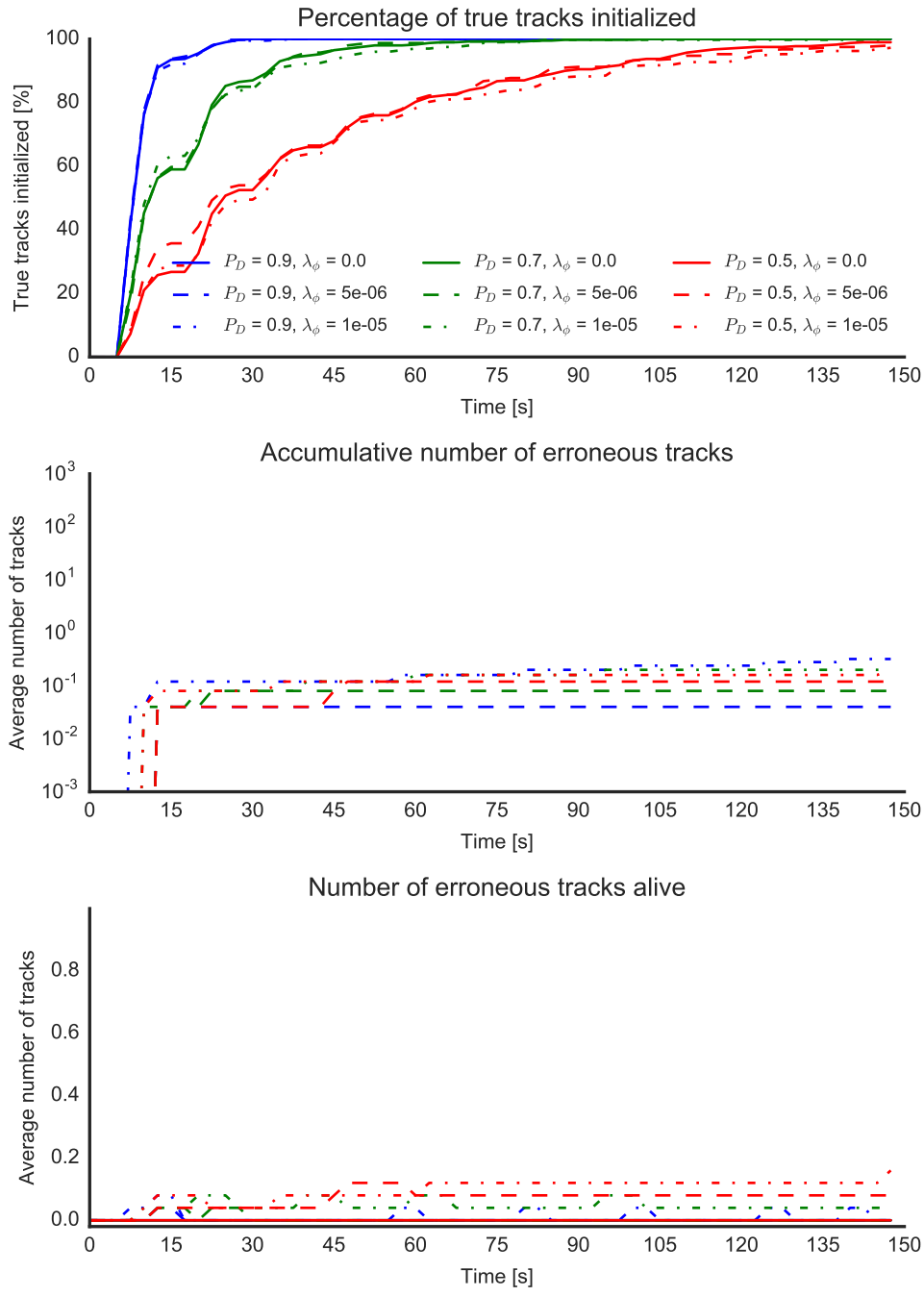

Figure B.58: Scenario 4 — Initialization time (3/4)

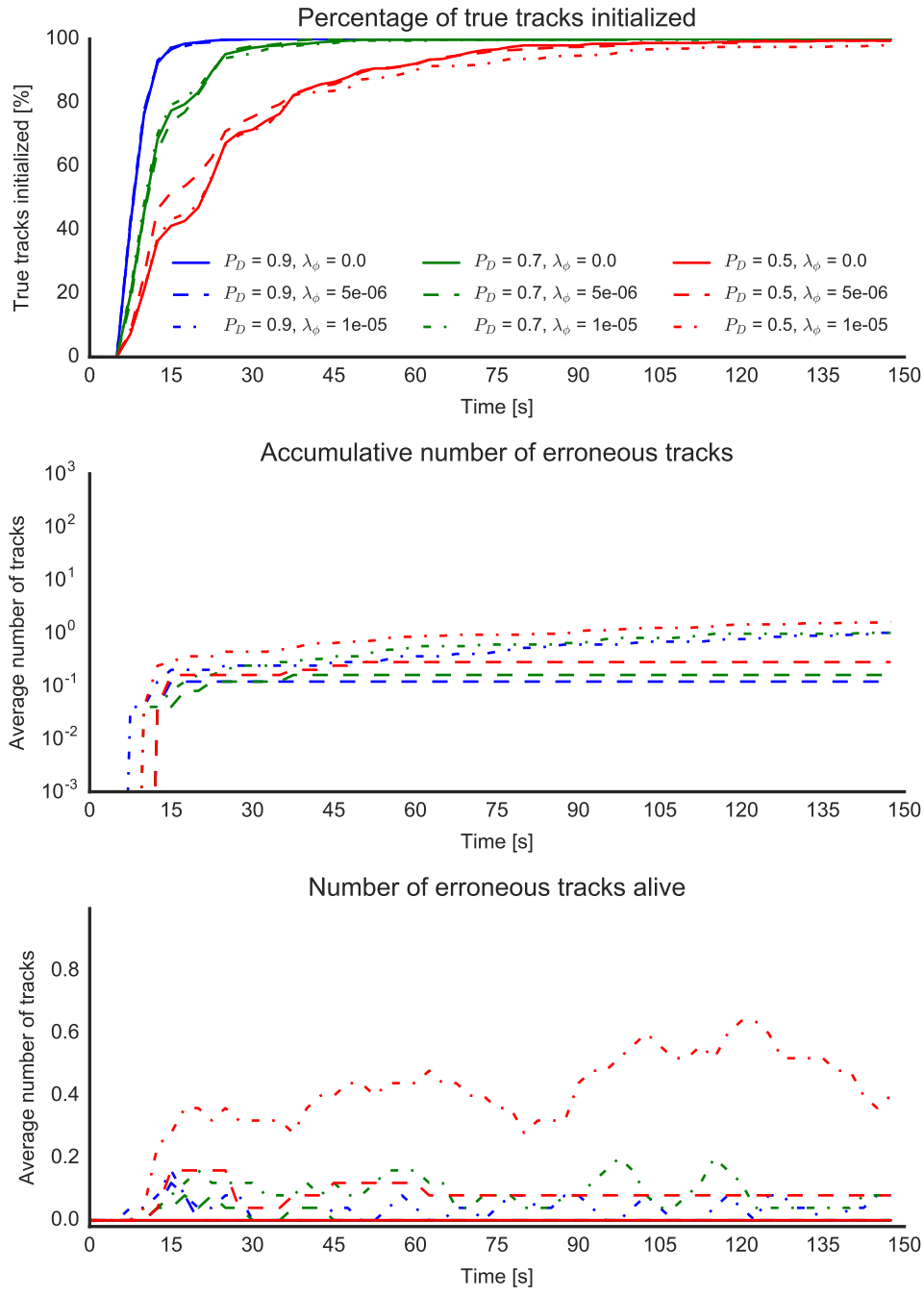

Figure B.59: Scenario 4 — Initialization time (3/5)

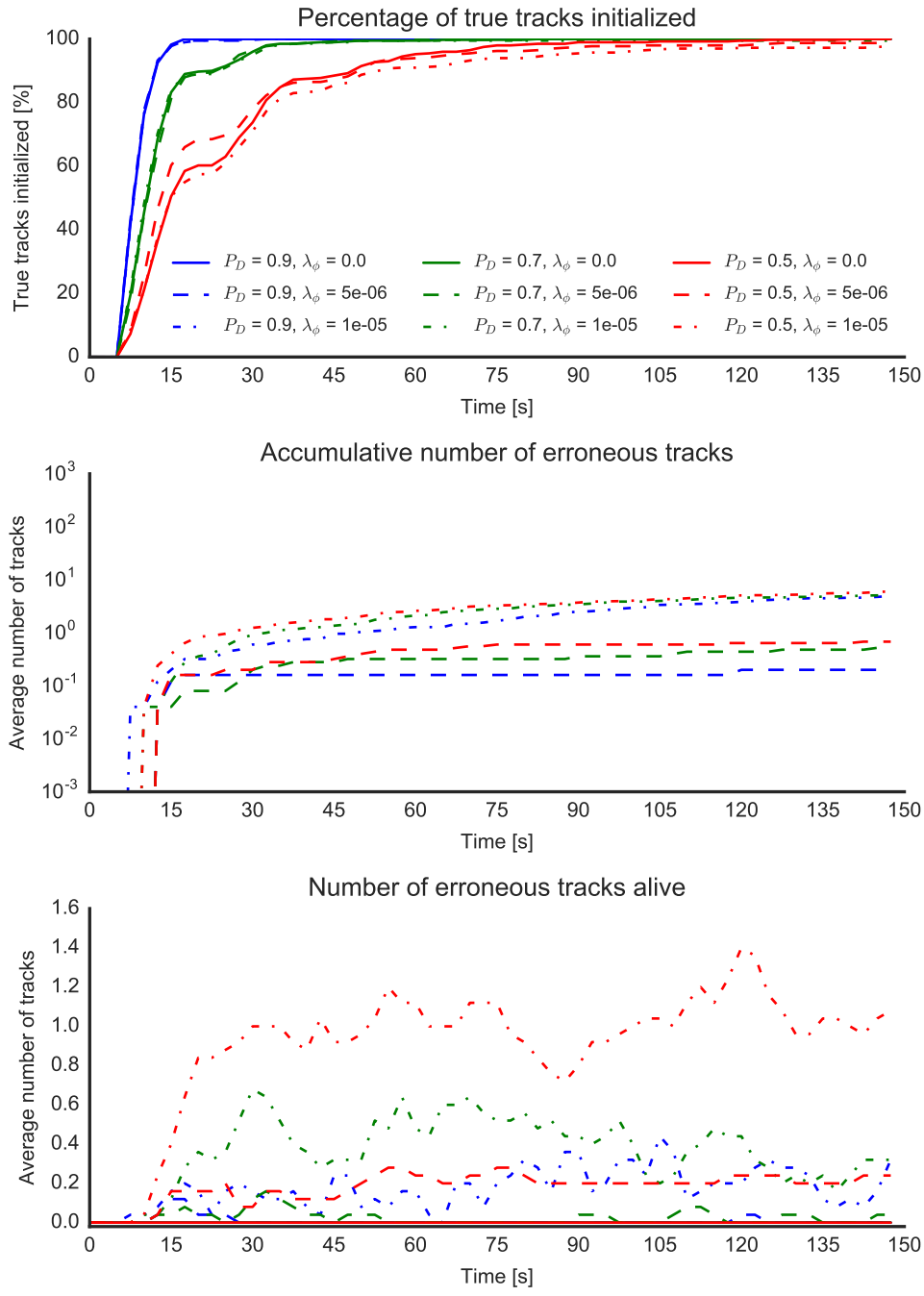

Figure B.60: Scenario 4 — Initialization time (3/6)

Bibliography

[1] I. B. Hagen, "Collision Avoidance for ASVs Using Model Predictive Control", Master's thesis, Norwegian University of Science and Technology, 2017.

[2] E. Liland, *An ILP approach to Multi Hypothesis Tracking*, Trondheim, 2017. [Online]. Available: osf.io/jp4pp.

[3] IALA, "IALA Guideline No. 1028 On The Automatic Identification", Tech. Rep. 1028, 2004, pp. 1–131.

[4] C. Carthel, S. Coraluppi, R. Grasso, and P. Grignan, "Fusion of AIS, RADAR, and SAR data for maritime surveillance", *Proc. of SPIE - The International Society for Optical Engineering*, 6748, no., 2007.

[5] C. Carthel, S. Coraluppi, and P. Grignan, "Multisensor tracking and fusion for maritime surveillance", *Proceedings of the 10th International Conference on Information Fusion*, no., 2007.

[6] S. Coraluppi, C. Carthel, M. Luettgen, and S. Lynch, "All-Source Track and Identity Fusion", in *DTIC Technical Reports (U.S. Defence Technical Information Center)*, 2000.

[7] M. T. Wolf, C. Assad, Y. Kuwata, A. Howard, H. Aghazarian, D. Zhu, T. Lu, A. Trebi-Ollennu, and T. Huntsberger, "360-degree visual detection and target tracking on an autonomous surface vehicle", *Journal of Field Robotics*, 27, no., pp. 819–833, 2010.

[8] P. Švec, A. Thakur, E. Raboin, B. C. Shah, and S. K. Gupta, "Target following with motion prediction for unmanned surface vehicle operating in cluttered environments", *Autonomous Robots*, 36, no., pp. 383–405, 2014.

[9] B.-N. Vo, M. Mallick, Y. Bar-Shalom, S. Coraluppi, R. Osborne, R. Mahler, and B.-T. Vo, "Multitarget tracking", *Wiley Encyclopedia of Electrical and Electronics Engineering*, no., 2015.

[10] Y. Bar-Shalom and T. E. Fortmann, "Algorithms for Tracking A Single Target In Clutter", in *Tracking and Data Association*, New York: Academic, 1998, pp. 119–185.

[11] H. A. P. Blom and E. Bloem, "Probabilistic Data Association Avoiding Track Coalescence", in *IEEE Transactions on Automatic Control*, vol. 45, 2000, pp. 247–259.

[12] P. Horridge and S. Maskell, "Real-time tracking of hundreds of targets with efficient exact JPDAF implementation", in *2006 9th International Conference on Information Fusion, FUSION*, 2006.

[13] R. J. Fitzgerald, "Development of Practical PDA Logic for Multitarget Tracking by Microprocessor", *American Control Conference, 1986*, no., pp. 889–898, 1986.

[14] R. Singer, R. Sea, and K. Housewright, "Derivation and evaluation of improved tracking filter for use in dense multitarget environments", *IEEE Transactions on Information Theory*, 20, no., pp. 423–432, 1974.

[15] D. Reid, "An algorithm for tracking multiple targets", *IEEE Transactions on Automatic Control*, 24, no., pp. 843–854, 1979.

[16] T. Kurien, "Issues in the design of practical multitarget tracking algorithms", in *Multitarget-Multisensor Tracking: Application and Advances*, First, Artech House, 1990, ch. 3, pp. 219–245.

[17] Y. Bar-Shalom, S. S. Blackman, and R. J. Fitzgerald, "Dimensionless score function for multiple hypothesis tracking", *IEEE Transactions on Aerospace and Electronic Systems*, 43, no., pp. 392–400, 2007.

[18] S. Blackman, "Multiple hypothesis tracking for multiple target tracking", *IEEE Aerospace and Electronic Systems Magazine*, 19, no., pp. 5–18, 2004. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1263228.

[19] S. S. Blackman, R. J. Dempster, and R. W. Reed, "Demonstration of Multiple Hypothesis Tracking (MHT) Practical Real-Time Implementation Feasibility", *Proceedings of SPIE Vol. 4473*, 4473, no., pp. 470–475, 2001.

[20] R. Mahler, "PHD filters of higher order in target number", *IEEE Transactions on Aerospace and Electronic Systems*, 43, no., pp. 1523–1543, 2007.

[21] M. Mahmuddin and Y. Yusof, "Automatic estimation total number of cluster using a hybrid test-and-generate and K-means algorithm", *2010 International Conference on Computer Applications and Industrial Electronics*, no., pp. 593–596, 2010.

[22] D. Pelleg and A. Moore, "X-means: Extending k-means with efficient estimation of the number of cluster", *Seventeeth International Conference on Machine Learning*, no., 2000.

[23] E. Brekke, O. Hallingstad, and J. Glattetre, "The signal-to-noise ratio of human divers", *OCEANS'10 IEEE Sydney*, no., pp. 1–10, 2010.

[24] Y. Bar-Shalom and X.-R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*. YBS Publishing, 1995.

[25] E. F. Wilthil, A. L. Flåten, and E. F. Brekke, "A Target Tracking System for ASV Collision Avoidance Based on the PDAF", in *Fossen, Thor I. Pettersen, Kristin Y. and Nijmeijer, Henk*, T. I. Fossen, K. Y. Pettersen, and H. Nijmeijer, Eds., Springer International Publishing, 2017, pp. 269–288.

[26] A. Harati-Mokhtari, A. Wall, P. Brooks, and J. Wang, "Automatic identification system (AIS): Data Reliability and Human Error Implications", *Journal of Navigation*, 60, no., pp. 373–389, 2007.

[27] C. H. Allen, *Farwells Rules Of The Nautical Road*, Eight Edit. Annapolis, MD: Naval Institute Press, 2005.

[28] IMO, "COLREGS - International Regulations for Preventing Collisions at Sea", *Convention on the International Regulations for Preventing Collisions at Sea, 1972*, no., pp. 1–74, 1972.

[29] E. Brekke, O. Hallingstad, and J. Glattetre, "Improved target tracking in the presence of wakes", *IEEE Transactions on Aerospace and Electronic Systems*, 48, no., pp. 1005–1017, 2012.

[30] H. Chen, T. Kirubarajan, and Y. Bar-Shalom, "Performance limits of track-to-track fusion versus centralized estimation: Theory and application", *IEEE Transactions on Aerospace and Electronic Systems*, 39, no., pp. 386–400, 2003.

[31] B. Habtemariam, R. Tharmarasa, M. McDonald, and T. Kirubarajan, "Measurement level AIS/radar fusion", *Signal Processing*, 106, no., pp. 348–357, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.sigpro.2014.07.029.

[32] J. Munkres, "Algorithms for the Assignment and Transportation Problems", *Journal of the Society for Industrial and Applied Mathematics*, 5, no., pp. 32–38, 1957.

[33] S. Chen, L. Liang, and Y. Tian, "The number of connected components in a graph associated with a rectangular (0,1)-matrix", *Linear Algebra and its Applications*, 487, no., pp. 74–85, 2015.

[34] R. G. Brown and P. Y. Hwang, *Introduction to random signals and applied Kalman filtering : with MATLAB exercises*. 2012, vol. 4th.

[35] D. Schuhmacher, B. T. Vo, and B. N. Vo, "On performance evaluation of multi-object filters", in *Proceedings of the 11th International Conference on Information Fusion, FUSION*, 2008.