5-2-2018

Prediction-based Opportunistic Sensing for Energy-Efficiency and Reliability in Distributed Sensor Networks

James Z. Hare
University of Connecticut, james.hare@uconn.edu

Prediction-based Opportunistic Sensing for Energy-Efficiency and Reliability in Distributed Sensor Networks

James Zachary Hare
University of Connecticut 2018

Distributed Sensor Networks (DSN) containing a large number of sensors nodes are rapidly advancing to perform automated tasks for a variety of applications (e.g. Target Tracking). One of the main problems studied in DSN is the Target Coverage problem, where the objective is to maximize the network lifetime while ensuring that all targets are covered during all times. Therefore, the main challenges in developing a DSN are network control strategies for energy-efficiency and resilience in the event of sensor failures. This thesis presents the *Prediction-based Opportunistic Sensing for Energy-efficiency* (POSE) algorithm, which aims to address these challenges by designing a DSN that minimizes energy consumption while ensuring target coverage. The first algorithm presents a distributed node-level supervisor that controls each sensor node to allow the network to self-adapt to the targets' trajectory via opportunistic sensing. This approach minimizes the network energy consumption by only enabling high power consuming devices when a target is predicted to travel within the sensor nodes coverage area, while using low power consuming devices when a target is absent. The second algorithm, *POSE using Distributed Classification, Clustering and Control* (POSE.3C), extends the POSE algorithm to include classification in the control loop to opportunistically observe targets of interest, while minimizing energy consumption via distributed clustering. Finally, the third algorithm, *POSE for Resilience* (POSE.R), extends the POSE.3C algorithm to incorporate Resilient Target Coverage into the DSN. This approach incorporates target and network density predictions to adapt the distributed clustering method to ensure that a target is covered in the event of sensor failures. The compilation of the three algorithms presents a distributed control strategy for DSN that performs energy-efficient and resilient target coverage.

Prediction-based Opportunistic Sensing for Energy-Efficiency and Reliability in Distributed Sensor Networks

James Zachary Hare

B.S., University of Connecticut, 2012
M.S., University of Connecticut, 2016

A Dissertation
Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy
at the
University of Connecticut

2018

Approval Page

Doctor of Philosophy Dissertation

Prediction-based Opportunistic Sensing for Energy-Efficiency and Reliability in Distributed Sensor Networks

Presented by

James Zachary Hare

Major Advisor

____

Prof. Shalabh Gupta

Associate Advisor

____

Prof. Peter Luh

Associate Advisor

____

Prof. Ashwin Dani

Associate Advisor

____

Dr. Thomas Wettergren

University of Connecticut

2018

Acknowledgments

Firstly, I would like to thank my advisor Prof. Shalabh Gupta for the continuous support over my Ph.D career. Prof. Gupta's patience, guidance, and motivation have helped me develop the research and work ethic skills that I carry today. I could not have finished my Ph.D study without his help, and I cannot thank him enough for all of the encouragement.

Next, I would like to thank Dr. Thomas Wettergren for providing insight on how to apply the control algorithm presented in this thesis to practical problems and applications. Dr. Wettergren was kind enough to have weekly discussions with me about my research and helped to develop the POSE algorithm into what is it today.

I would also like to thank my advisory committee: Prof. Peter Luh and Prof. Ashwin Dani, for their insightful comments and difficult questions during my thesis defense that have helped to improve the presentation of my research.

I thank my fellow labmates and peers for the stimulating discussions, hours of brainstorming, and support throughout my Ph.D career. An engineers research progression can be severely stunted without the encouragement, questions, and unique perspectives that their peers provide. I cannot thank them enough for providing their time and effort into helping my research improve.

Last but not least, I would like to thank my family and friends for all of their support in completing the Ph.D program. My family provided me with the opportunity and support to pursue a graduate degree at the University of Connecticut, and without their encouragement, I wouldn't be in this position today. Finally, my friends have provided tremendous amounts of encouragement and comfort throughout my Ph.D career and without them, I may not have proceeded to where I am today.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF NOMENCLATURE

Acronyms

CH	Cluster Head
DPU	Data Processing Unit
DSN	Distributed Sensor Network
DWNA	Discrete White Noise Acceleration
EGDOP	Energy-based Geometric Dilution of Precision
GDOP	Geometric Dilution of Precision
GPS	Global Positioning System
HPS	High Power Sensing
JPDA	Joint Probabilistic Data Association
LPS	Low Power Sensing
PFSA	Probabilistic Finite State Automaton
PIR	Passive Infrared Sensor
POSE	Prediction-based Opportunistic Sensing

POSE.3C	Prediction-based Opportunistic Sensing using Distributed Classification, Clustering and Control
POSE.R	Prediction-based Opportunistic Sensing for Resilient Target Coverage and Tracking
ROI	Region Of Interest
RX	Receiver
T2TA	Track-2-Track Association
T2TF	Track-2-Track Fusion
TNOI	Target Not Of Interest
TOI	Target Of Interest
TX	Transmitter

Symbols

α	Detection probability
$\mathcal{A}$	Set of actions
A	PFSA alphabet
a_i	Action of sensor s_i
A_{Ω_1}	Area of region Ω_1
A_{Ω_2}	Area of region Ω_2
A_{Ω_3}	Area of region Ω_3
A_{Ω}	Area of Region Of Interest
β	Decay rate of detection probability with distance
B	Confusion Matrix
ΔT	Sampling time interval

$\hat{D}^{s_i}(k)$ Target class decision of node s_i

$\mathcal{S}_{c,R_1}(\mathbf{u}^{\tau_\ell}(k))$ Set of nodes that are covering target τ_ℓ with HPS sensing range R_1

$D(\mathbf{u}^{\tau_\ell}(k))$ Coverage degree at time k

$D_{R_1}(\mathbf{u}^{\tau_\ell}(k))$ Base Coverage Degree

$\bar{E}_{\Delta T}$ Expected energy consumption of the POSE.3C network

$\mathbf{E}_{det}$ Set of energy remaining of candidate sensors

$E^{s_i}(k)$ Total Energy consumed by node s_i until time k

$e_j^{s_i}$ Rate of energy consumption per unit time of a certain device $j \in \{$DPU, LPS, HPS, transmitter, receiver, Clock$\}$

E_0 Initial energy of sensor nodes

E_{HPS} Energy consumption of the *HPS* state

$E_{net}(k)$ Total energy consumed by the entire network until time k

$E_R^{s_i}(k)$ Node s_i's percent of energy remaining at time k

E_{Sleep} Initial energy of sensor nodes

$\mathbf{f}(\mathbf{x}(k), k)$ State transition matrix at time k

f False alarm rate

γ_{HPS} HPS device detection probability

$G(k)$ Coverage gap

$\mathbf{h}(\mathbf{x}(k), k)$ Measurement model

$\hat{\mathbf{I}}^{s_i}(k)$ Received information ensemble

$\hat{\mathbf{I}}_T^{s_i,c}(k)$ Associated information ensemble for target c

$\hat{\mathbf{I}}_T^{s_i}(k)$ Trustworthy information ensemble

λ	Number of targets in Ω_γ during every time step of the network's lifetime
$\Lambda^{s_i,c}(k+1)$	Probability of target c traveling through node s_i's HPS device coverage area at time $k+1$
L	Length of tube Ω_γ
μ_{cl}	False measurement mean
M	Number of time steps that must have a measurement associated to a track for validation
m	Number of targets traveling through Ω
$\mathcal{N}^{s_i}$	Node s_i's set of neighbors
$\mathcal{N}^{s_i}_{HPS}$	Set of *HPS* state nodes that have broadcasted target information
$\mathcal{N}^{s_i}_{T}$	Set of trustworthy neighbors
N	Number of overall time steps to consider if a track is valid
n	Number of multi-modal sensor nodes in Ω
N'_{sel}	Number of filtered sensors used for sensor selection
N_{sel}	Desired coverage degree
N_{sel}	Desired number of sensors to track the target in the *HPS* state
Ω	Region Of Interest
$\Omega^{s_i}(k)$	Node s_i's coverage area at time k
Ω_1	Detection region around the target
Ω_1^*	Selected sensors in Ω_1
Ω_2	Region outside of the detection region but within the communication region around the target
Ω_3	Region away from the target

Ω_G	Target coverage area
Ω_γ	Tube with the highest frequency of targets traveling through the network
Ω_{can}	Candidate region
$\Omega_{v_{j,h}}$	Cell of the target coverage area Ω_G
ϕ_{s_i}	Azimuth angle between node s_i and the targets predicted position
Θ	Set of PFSA states
θ_i	PFSA state i
Ξ	Probabilistic finite state automaton
Φ	Potential function
$\psi(k)$	Target's turning rate at time k
P	PFSA state transition matrix
P_m	Probability of missed detection
$P_{D,HPS}^{s_i}(\mathbf{u}^{\tau_\ell}(k))$	HPS device detection probability
$P_{D,LPS}^{s_i}(\mathbf{u}^{\tau_\ell}(k))$	Low Power Sensing devices probability of detection
p_{drop}	Probability of dropping packets
p_{fa}	Probability of false alarm
$P_{HPS}^{s_i}(k)$	Probability of detecting the target with the HPS device
$p_{i,j}$	PFSA state transition probability from state θ_i to state θ_j
p_{rand}	Probability of sleeping in a Random Network
$P_{RX}^{s_i}(k)$	Probability of receiving packets
p_{sleep}	Probability of staying in the *Sleep* state

$\mathbf{Q}$	Process noise covariance
$\mathcal{N}_{B_{j,h}}$	Number of nodes covering cell $\Omega_{v_{j,h}}$
$\mathcal{R}^*$	Set of optimal sensing ranges
$\mathcal{S}_{det}$	Set of candidate sensors
$\mathbf{R}$	Measurement noise covariance
ρ	Network density
$B_{j,h}$	Coverage function
R_c	Communication radius of transmitter/receiver
R_{HPS}	High power sensing device's sensing range
$R^{s_i}_{HPS}(k)$	Node s_i's HPS range at time k
R_{LPS}	Low power sensing device's sensing range
R_r	Low power sensing device's reliable sensing range
r_{s_i}	Range of sensor s_i to the target's predicted position
$\mathcal{S}$	Set of sensor nodes in Region of Interest
$\mathcal{S}^*$	Set of optimal sensors
$\mathcal{S}_c(\mathbf{u}^{\tau_\ell}(k))$	Set of nodes that contain target τ_ℓ within their coverage areas at time k
$\mathcal{S}_E$	Set of energy-based ranking sensors
$\mathcal{S}_{det}$	Set of candidate sensor nodes
σ_ϕ	Standard deviation of azimuth measurement
σ_ψ	Process noise of the turning rate ψ
σ_R	Standard deviation of range measurement

σ_x	Process noise of x position
σ_y	Process noise of y position
$\bar{T}_{Life}$	Network lifetime
$\mathcal{T}$	Set of targets in Region of Interest
$\mathcal{U}$	Node utility function
$\mathbf{u}^{\tau_\ell}(k)$	Location of target τ_ℓ at time k
$\mathbf{u}^{s_i}$	Location of node s_i
$\boldsymbol{v}(k)$	Process noise at time k
$v_{i,j}$	Worth of cell $\Omega_{v_{j,h}}$
$\hat{\mathbf{W}}^{s_i}(k)$	Node s_i's filter gain
$\mathbf{w}(k)$	Measurement noise
W	Sum of the worth gained in the potential game
$\chi_j^{s_i}(k)$	Indicator of whether device $j \in \{DPU, LPS, HPS, transmitter, receiver, clock\}$ is ON of OFF at time k
$\dot{x}(k)$	Target's x velocity at time k
$\mathbf{x}(k)$	Target state at time k
$x(k)$	Target's x position at time k
$\dot{y}(k)$	Target's y velocity at time k
$y(k)$	Target's y position at time k
$\mathbf{z}(k)$	Set of measurements collected by HPS device

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

With the recent developments in semiconductor devices, data processing units, sensor technologies, and communication systems, Distributed Sensor Networks (DSN) are evolving to perform many useful tasks in land, air, and sea for a variety of applications. For example, DSN are being developed to gather intelligence [1–6]; perform surveillance [7, 8]; monitor wildlife habitats [9, 10], potential disasters [11, 12], personal well being [13, 14], traffic conditions [15]; and many other applications. These networks have evolved to consist of intelligent sensor nodes that each integrate multiple heterogeneous sensing devices, a transceiver, a data processing unit, and a memory unit together [16]. Additionally, they are envisioned to operate autonomously to achieve the desired network objective. This thesis studies the *Target Coverage* problem, which aims to maximize the network lifetime while maintaining tracking performance. This requires that the network is controlled in a resilient and energy-efficient manner. A summary of the objective, challenges, solution approaches, and research gaps are presented in Fig. 1.1 and discussed next.

The intelligent sensor nodes are typically deployed randomly throughout a *Region Of Interest* (ROI) and are envisioned to track targets for long periods of time. However, one major limitation is the limited availability of energy resources on-board each sensor node [17]. If a few sensor

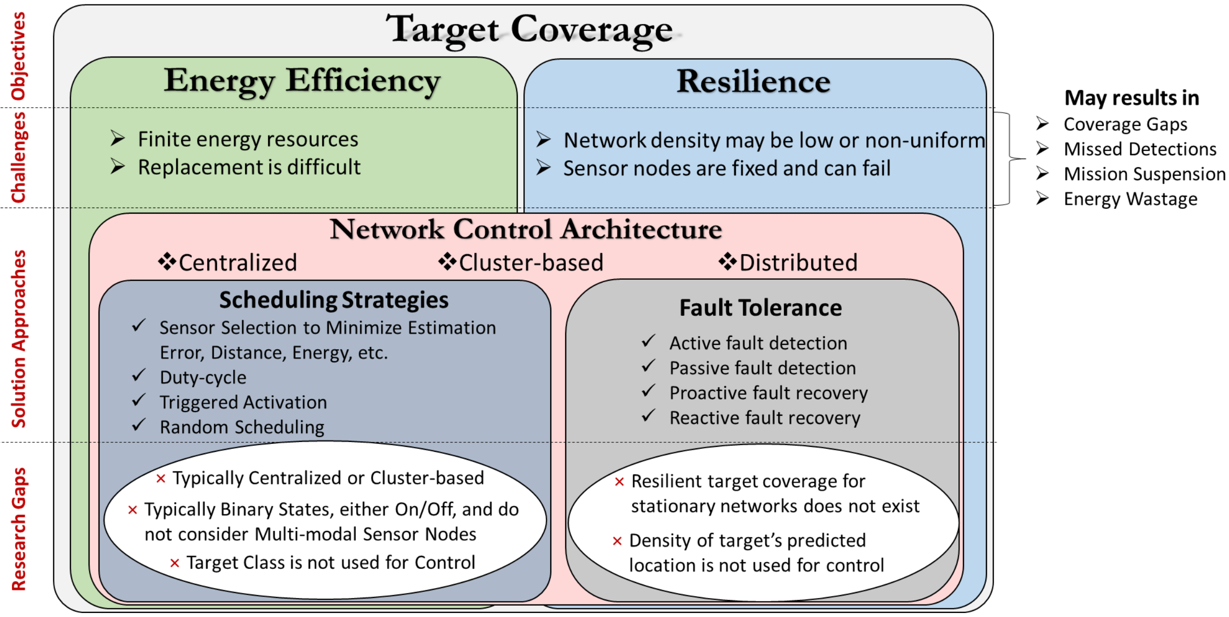

Objectives
Challenges
Solution Approaches
Research Gaps
Target Coverage
Energy Efficiency
Resilience
Finite energy resources
Replacement is difficult
Network density may be low or non-uniform
Sensor nodes are fixed and can fail
May results in
Coverage Gaps
Missed Detections
Mission Suspension
Energy Wastage
Network Control Architecture
Centralized
Cluster-based
Distributed
Scheduling Strategies
Sensor Selection to Minimize Estimation Error, Distance, Energy, etc.
Duty-cycle
Triggered Activation
Random Scheduling
Fault Tolerance
Active fault detection
Passive fault detection
Proactive fault recovery
Reactive fault recovery
Typically Centralized or Cluster-based
Typically Binary States, either On/Off, and do not consider Multi-modal Sensor Nodes
Target Class is not used for Control
Resilient target coverage for stationary networks does not exist
Density of target's predicted location is not used for control

Figure 1.1: Challenges and Current Approaches of the Target Coverage Problem

nodes deplete their energy, they fail to collect data, causing missed detections and coverage gaps [18]. Often, it is difficult and time consuming to charge or replace the depleted nodes due to the large size and geographical region of deployment. This results in poor network performance, information delays, equipment loss, and mission failures. Therefore, it is important that the sensor nodes are controlled in an energy-efficiency manner to improve the network lifetime.

Energy-efficient control strategies presented in literature are typically achieved using one of the following network control architectures: (i) Centralized, (ii) Cluster-based, and (iii) Distributed. Centralized networks require each sensor node to communicate their observations to a central computing station, which calculates and transmits the optimal scheduling decisions to each sensor node while estimating the target's trajectory [19]. Although centralized architectures can provide the optimal solution, these networks are very difficult to implement on large scale network due to multiple factors, such as communication limitations, transmission delays, and data processing requirements. To overcome these limitations, cluster-based networks were proposed. These networks divide the ROI into several disjoint clusters, where each cluster contains a node that acts as a central computer, called a *cluster head* (CH) [20]. The CH receives target measurements from nodes within the cluster, computes the scheduling actions, performs state estimation, and relays the actions to each node. Although this improves the limitations of centralized schemes, data trasmission between sensor nodes and the CH occur frequently [21] and may lead to energy wastage. Furthermore, if a CH were to fail, due to defects or energy depletion, the entire cluster would stop recieving scheduling decisions, rendering a large gap in the coverage area [22]. This requires implementation of dynamic CH election schemes [23].

Distributed network architectures have also been proposed where each sensor node dynamically controls its own tasks based on the information measured by its own sensor suite or perceived by its neighboring nodes [24]. Unlike cluster-based architectures, the distributed networks tend to be reactive in the sense that they only communicate information when an event has occurred, thus allowing for savings on transmission costs. Additionally, distributed architectures do not require sensor or CH selection, thus allowing for less computation requirements.

Within the three network architectures, energy-efficient control is achieved through sensor scheduling. These methods are categorized as either (i) Passive or (ii) Active strategies. Passive strategies are typically set up offline where the sensor nodes operate in an *On* or *Off* state based on a fixed or probabilistic manner [25]. Duty-cycle scheduling consists of fixing a time interval for each operating condition (On/Off), whereas Random scheduling has each sensor node stay in a particular task for a fixed time interval and transition to a new task based on a predefined probability. Active scheduling strategies utilize information about the target or the network topology to adjust the tasks of each sensor node. One approach to active scheduling is to utilize target detections to activate a cluster of sensors, known as Trigger-based Activation [26]. This approach triggers the nodes On once a target is detected within their local area. A more efficient approach to active scheduling utilizes target information to optimally select the best nodes to be On and cover the target while conserving energy by turning the remaining nodes Off using a Sensor Selection method. These methods typically consist of maximizing/minimizing one of the following cost functions: probability of detection [27], tracking accuracy, [28, 29], energy [30], Kullback-Liebler distance [31], or other information theoretic measures [32].

Although the above scheduling methods achieve the desired goal of energy-efficient target coverage, there are still some research gaps that were not addressed in their work. First, the above scheduling methods are typically for centralized or cluster-based network architectures. There are a few distributed techniques available, but most of the active scheduling methods are cluster-based. Second, the scheduling methods proposed consider binary operating conditions (On or Off) and do not consider multi-modal sensor nodes. When only considering binary operating conditions, energy consumption can significantly increase due to frequent communication or frequent use of active sensing devices. Finally, the proposed scheduling methods do not consider the type of target present in the control algorithm. In practical applications, not all of the targets that pass through the ROI will be of interest to the network operator. Therefore, if the network does not adapt to the type of target present, significant energy may be wasted.

Another critical challenge in a DSN is maintaining target coverage in the event of sensor node failures. Sensor nodes are prone to failures due to component degradation, hardware failure, malicious attacks, or battery depletion [33], which may cause changes to the network topology. If multiple co-located sensor nodes fail, a sector of the network may be uncovered, causing missed detections when a target travels through the coverage gap. Furthermore, the sensor nodes may be non-uniformly distributed causing high and low density regions, and even coverage gaps, due to geographical conditions and uncertainties [34]. Therefore, the development of a self-adaptive network that mitigates the formation of coverage gaps is necessary to ensure network resilience.

Handling sensor failures is typically achieved in literature by developing a fault tolerant network. This process consists of (i) fault detection and (ii) fault recovery methods. Fault detection methods are typically classified as either active or passive. Active fault detection is achieved using a centralized or cluster-based network topology [35] and consists of requesting constant updates throughout the network or by utilizing heartbeat signals. Passive monitoring is achieved in centralized, cluster-based, or distributed network architectures by observing the traffic already present in the network to infer the nodes health [36]. Although fault detection is a critical problem in sensor networks, fault recovery is essential to ensure target coverage.

Fault recovery methods are typically classified into two approaches, proactive and reactive. Most proactive approaches handle the problem preemptively by deploying redundant sensor nodes to acheive κ-coverage [37], where κ is the number of nodes that can cover a point/target. This approach allows for $\kappa - 1$ nodes to fail in any given area before target coverage is lost. However, this requires significantly more sensor nodes to be deployed, which can increase mission cost. Additionally, once $\geq \kappa$ co-located nodes fail, proactive fault recovery methods fail to meet the network objective. Reactive methods utilize fault detection information to plan a control action to fill in the hole created by nodes failures. To fill in the hole, most networks utilize mobile sensor nodes [33] or redeploy new sensor nodes [38]. For stationary sensor networks without the ability to redeploy or utilize mobile sensor nodes, only single sensor failure recovery methods have been proposed and mainly considering data recovery [39] and connectivity [40].

Thus, the following research gaps are present. First, resilient target coverage for stationary networks does not exist. Although proactive fault recovery methods are robust to sensor failures, they are not resilient to κ failures in a local area. Therefore, if a target travels into a coverage gap, the network will not achieve target coverage. Second, a distributed learning mechanism that identifies low density regions and coverage gaps around the target's trajectory does not exist. While passive methods for fault detection exist, none of the proposed methods are formulated to identify the density of the region that the target is traveling into. If the sensor nodes are aware that the target is traveling within a coverage gap, an adaptive control decision could be made to ensure target coverage.

Therefore, the main objective of this thesis is to develop a target coverage control strategy for a DSN that is energy-efficient and resilient. The following section gives an overview of the three themes that achieve the primary objective, while their details are presented in Chapters 3, 4, 5.

1.2 Outline and Contributions

This thesis presents a DSN control algorithm for multi-modal sensor nodes that ensures target coverage and extends the network's lifetime via *Opportunistic Sensing*. The objective of opportunistic sensing is to select and activate sensors only in the local region around an event. It is further desired to do so in a predictive manner to pro-actively prepare for the event in advance. In this regard, this thesis is split into three themes that allow for energy-efficient control and resilient target coverage.

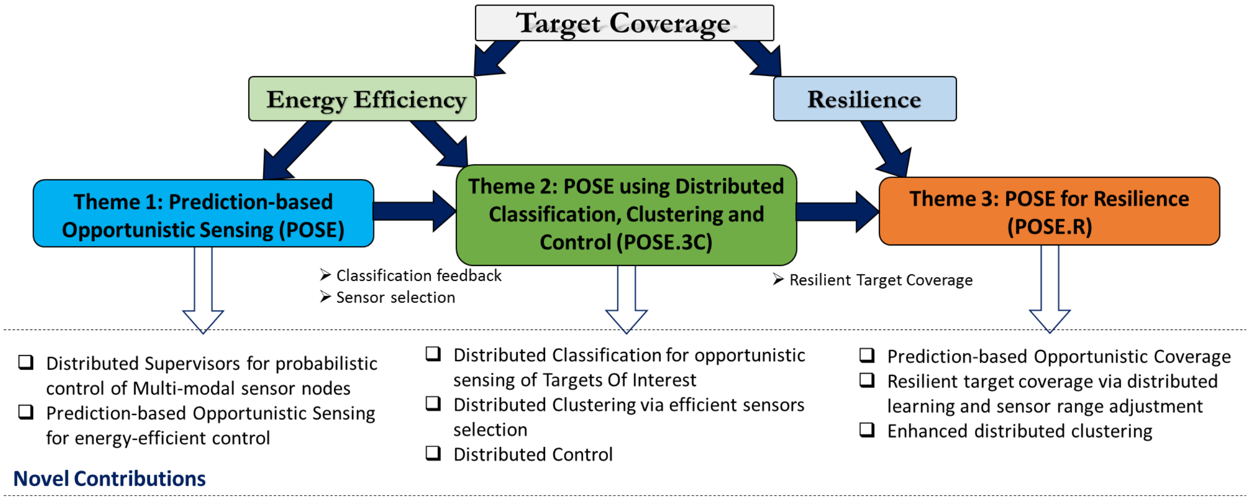

Target Coverage
Energy Efficiency
Resilience
Theme 1: Prediction-based Opportunistic Sensing (POSE)
Theme 2: POSE using Distributed Classification, Clustering and Control (POSE.3C)
Theme 3: POSE for Resilience (POSE.R)
➢ Classification feedback
➢ Sensor selection
➢ Resilient Target Coverage
❑ Distributed Supervisors for probabilistic control of Multi-modal sensor nodes
❑ Prediction-based Opportunistic Sensing for energy-efficient control
❑ Distributed Classification for opportunistic sensing of Targets Of Interest
❑ Distributed Clustering via efficient sensors selection
❑ Distributed Control
❑ Prediction-based Opportunistic Coverage
❑ Resilient target coverage via distributed learning and sensor range adjustment
❑ Enhanced distributed clustering
Novel Contributions

Figure 1.2: Outline of the Proposed Thesis Themes

A summary of the three themes and their contributions are presented in Figure 1.2.

1.2.1 Theme 1: Prediction-based Opportunistic Sensing

The first theme of this thesis address the problem of energy-efficiency in DSN by developing the *Prediction-based Opportunistic Sensing* (POSE) algorithm [41]. This algorithm manages the power consumption of each sensor node in a distributed fashion to enable energy-efficient target coverage. Each multi-modal sensor node consists of multiple devices including high power sensors, low power sensors, a data processing unit and a communication device. Each node also contains an embedded distributed supervisor, i.e. a *Probabilistic Finite State Automaton* (PFSA), which enables and disables the devices on the node in an opportunistic manner to conserve energy.

The states of the PFSA represent the different activities of the node which correspond to enabling/disabling its various sensing and communication devices [8]. The state transition probabilities of the PFSA are dynamically updated based on the information received from neighboring nodes or from direct observation of a target using the local sensor suite. The probabilities control the switching of the PFSA states to facilitate opportunistic sensing based on the predicted target(s) trajectories. For example, the PFSA of a sensor node enables its high power sensing devices when a target is present within its coverage area, while disabling them and entering into a low power consuming cycle when it is not present.

The novel contributions of the first theme are as follows:

- The development of distributed supervisors (i.e. PFSA) for probabilistic control of the devices enabled on each multi-modal sensor node, and
- A prediction-based scheduling approach that updates the probabilities of the distributed supervisor to enable opportunistic sensing for energy-efficient control.

1.2.2 Theme 2: Prediction-based Opportunistic Sensing using Distributed Classification, Clustering and Control

The POSE algorithm allows for significant energy savings compared with distributed scheduling methods while achieving similar missed detection rates as compared to a network that is always monitoring. However, the POSE algorithm has each sensor node enable its high power sensing devices when a target is present within its coverage area. This may result in redundant sensor nodes tracking the target if the network density is high or the sensing radius is large. Additionally, in practical applications, all of the targets within the field may not be of interest to the network observer. This results in energy wastage due to sensor nodes tracking the targets not of interest.

To address the above issues, the *Prediction-based Opportunistic Sensing using Distributed Classification, Clustering and Control* (POSE.3C) algorithm was developed as the second theme of this thesis to further improve the energy-efficiency of the DSN [42]. The POSE.3C algorithm continues using the distributed supervisor (i.e. PFSA) developed in the POSE algorithm, except the state transition probabilities are updated based on the target's class as well as the target's predicted location. This new algorithm is composed of two additional steps in the control scheme, 1) Target classification and 2) Sensor selection.

The sensor nodes that are tracking the target perform target classification to identify if the target is a *Target Of Interest* (TOI) or a *Target Not Of Interest* (TNOI). For example, in a boarder surveillance application, the TOIs could be humans and vehicles, while the TNOIs could be animals. Then, the sensor nodes that use received information to update the PFSA will perform distributed fusion to identify the type of target present. In this algorithm, a distributed clustering approach is employed for sensor selection to select $N_{sel} > 1$ sensor nodes to track TOI to improve estimation, while only selecting $N_{sel} = 1$ sensor nodes to track TNOI to maintain awareness. In this regard, the DSN opportunistically senses TOI via classification feedback; minimizes energy wastage through distributed clustering; and extends the network lifetime using distributed supervisors for control.

The novel contributions of the second theme are as follows:

- Development of a distributed algorithm that facilitates 3C network autonomy with the following attributes:
  (i) distributed classification and utilization of the target class information to govern the cluster size for tracking,
  (ii) distributed clustering via a) fusing the target state estimates and predicting the next state, and b) sensor node selection around the target's predicted state, and
  (iii) distributed supervisory control to enable/disable multi-modal sensing devices for energy-efficiency.
- Theoretical analysis of the network's performance characteristics: (i) Expected energy consumption, (ii) Expected network lifetime, (iii) Probability of missed detection for a target birth and for a mature target.

1.2.3 Theme 3: Prediction-based Opportunistic Sensing for Resilient Target Coverage and Tracking

The POSE and POSE.3C algorithms improve the energy-efficiency of the DSN, however, they do not address the problems that occur when sensor nodes fail. Therefore, the third theme of this thesis further extends the first and second themes by incorporating resilient target coverage into the DSN by developing the *Prediction-based Opportunistic Sensing for Resilient Target Coverage and Tracking* (POSE.R) algorithm [43]. Here, the multi-modal sensor nodes are assumed to contain a high power sensing device that can vary it's sensing range at the cost of consuming more energy. The POSE.R algorithm extends the POSE and POSE.3C algorithms by incorporating an adaptive sensor selection approach that selects the optimal sensor nodes and their sensing ranges to track the target and ensure target coverage in the presence of failures and non-uniform node locations. The sensor selection approach adapts to the density of active sensor nodes around the target's predicted location. For high density regions with $\geq N_{sel}$ nodes, an *Energy-based Geometric Dilution of Precision* (EGDOP) sensor selection method is employed to select and activate geometrically diverse nodes with high remaining energy to cover the target with their minimum sensing range.

When a target is predicted to travel through a low density region, the sensor nodes may need to expand their sensing range to accommodate for a coverage gap or an insufficient number of nodes. To identify the best sensing ranges to cover the target while minimizing energy consumption, a Game Theoretic sensor range selection method is employed using *Potential Games*. The objective function of the game selects N_{sel} nodes and their sensing ranges to cover the target's prediction location while minimizing energy wastage by reducing redundant coverage. In this regard, the DSN becomes resilient to coverage gaps by opportunistically covering the gaps when a target is predicted to travel through it, while operating in an energy-efficient manner when the target travels through high density regions.

The novel contributions of the third theme is the development of a distributed algorithm that facilitates resilient target coverage and tracking with the following attributes:

- distributed coverage gap identification method that does not rely on active or passive monitoring methods,

- distributed node selection approach that adapts to the network density around a target's predicted location via a) EGDOP selection criteria for high density regions, and b) a Game Theoretic node and range selection method using potential games for low density regions.

- distributed supervisory control strategy to enable/disable multi-modal sensing devices for energy-efficiency.

1.3 List of Publications

Journal Papers

1. **James Zachary Hare**, Shalabh Gupta, Junnan Song, and Thomas Wettergren. "POSE.R: Prediction-based Opportunistic Sensing for Resilient and Efficient Sensor Networks". In: *ACM Transactions on Sensor Networks,* ***Ready for submission*** (2018).

2. **James Zachary Hare**, Shalabh Gupta, and Thomas Wettergren. "POSE.3C: Prediction-based Opportunistic Sensing using Distributed Classification, Clustering and Control - A Network-centric Autonomy Concept". In: *IEEE Transactions on Control of Network Systems,* ***Under Review*** (2017).

3. **James Zachary Hare**, Shalabh Gupta, and Thomas Wettergren. "Prediction-based Opportunistic Sensing for Energy-efficient Sensor Networks using Distributed Supervisors". In: *IEEE Transactions on Cybernetics* (2017). DOI: `10.1109/TCYB.2017.2727981`.

4. Nayeff Najjar, Shalabh Gupta, **James Zachary Hare**, Sherif Kandil, and Rhonda Walthall. "Optimal Sensor Selection and Fusion for Heat Exchanger Fouling Diagnosis in Aerospace Systems". In: *IEEE Sensors Journal* 16.12 (2016), pp. 4866–4881.

5. **James Zachary Hare**, Xiaofang Shi, Shalabh Gupta, and Ali Bazzi. "Fault diagnostics in smart micro-grids: A survey". In: *Renewable and Sustainable Energy Reviews* 60 (2016), pp. 1114–1124.

6. Nayeff Najjar, Chaitanya Sankavaram, **James Zachary Hare**, Shalabh Gupta, Krishna Pattipati, Rhonda Walthall, and Paul D'Orlando. "Health Assessment of Liquid Cooling System in Aircrafts: Data Visualization, Reduction, Clustering and Classification". In: *SAE International Journal of Aerospace* 5.1 (2012), pp. 119–127.

Conference Papers

7. **James Zachary Hare**, Shalabh Gupta, Junnan Song, and Thomas Wettergren. "Classification induced distributed sensor scheduling for energy-efficiency in underwater target tracking sensor networks". In: *OCEANS Anchorage*. Sept. 2017.

8. Weiqiang Chen, Ali Bazzi, **James Zachary Hare**, and Shalabh Gupta. "Real-time integrated model of a microgrid with distributed clean energy generators and their power electronics". In: *IEEE Applied Power Electronics Conference and Exposition (APEC)*. 2016, pp. 2666–2672.

9. **James Zachary Hare**, Shalabh Gupta, Nayeff Najjar, Paul D'Orlando, and Rhonda Walthall. "System-Level Fault Diagnosis with Application to the Environmental Control System of an Aircraft". In: *SAE AeroTech Congress and Exhibition*. 2016.

10. **James Zachary Hare**, Shalabh Gupta, and James Wilson. "Decentralized smart sensor scheduling for multiple target tracking for border surveillance". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2015, pp. 3265–3270.

11. James Wilson, Nayeff Najjar, **James Zachary Hare**, and Shalabh Gupta. "Human activity recognition using LZW-coded probabilistic finite state automata". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2015, pp. 3018–3023.

12. Junnan Song, Shalabh Gupta, and **James Zachary Hare**. "Game-theoretic cooperative coverage using autonomous vehicles". In: *Oceans'14 MTS/IEEE*. 2014.

13. Junnan Song, Kaixiang Qiu, Shalabh Gupta, and **James Zachary Hare**. "SLAM based adaptive navigation of AUVs for oil spill cleaning". In: *Oceans'14 MTS/IEEE*. 2014.

14. **James Zachary Hare**, Shalabh Gupta, and Junnan Song. "Distributed smart sensor scheduling for underwater target tracking". In: *Oceans'14 MTS/IEEE*. 2014.

15. **James Zachary Hare**, Xiaofang Shi, Shalabh Gupta, and Ali Bazzi. "A review of faults and fault diagnosis in micro-grids electrical energy infrastructure". In: *IEEE Energy Conversion Congress and Exposition (ECCE)*. 2014, pp. 3325–3332.

16. Junnan Song, Shalabh Gupta, **James Zachary Hare**, and Shengli Zhou. "Adaptive cleaning of oil spills by autonomous vehicles under partial information". In: *Oceans'13 MTS/IEEE*. 2013.

17. Nayeff Najjar, **James Zachary Hare**, Andre Silva, Shalabh Gupta, Gregory Leaper, Krishna Pattipati, Rhonda Walthall, and Paul D'Orlando. "Heat Exchanger Fouling Diagnosis for an Aircraft Air-Conditioning System". In: *SAE AeroTech Congress and Exhibition*. 2013.

18. Shalabh Gupta, **James Zachary Hare**, and Shengli Zhou. "Cooperative coverage using autonomous underwater vehicles in unknown environments". In: *Oceans'12 MTS/IEEE*. 2012.

19. Nayeff Najjar, Chaitanya Sankavaram, **James Zachary Hare**, Shalabh Gupta, Krishna Pattipati, Rhonda Walthall, and Paul D'Orlando. "Health Assessment of Liquid Cooling System in Aircrafts: Data Visualization, Reduction, Clustering and Classification". In: *SAE Aerospace Electronics and Avionics Systems Conference*. 2012.

Patents

20. **James Zachary Hare**, Nayeff Najjar, and Shalabh Gupta. “System Level Fault Diagnosis on Aerospace Air Management Systems”. US Patent Application Number 14687112.

21. Nayeff Najjar, **James Zachary Hare**, and Shalabh Gupta. “Optimal Sensor Selection and Fusion for Heat Exchanger Fouling Diagnosis in Aerospace Systems”. US Patent Application Number 14700769.

CHAPTER 2

PROBLEM FORMULATION

Let $\Omega \subset \mathbb{R}^2$ be the Region Of Interest (ROI) with area A_Ω. Let $\mathcal{T} = \{\tau_1, \tau_2, ...\tau_m\}$ be the set of m targets traveling through Ω. Let $\mathcal{S} = \{s_1, s_2, ...s_n\}$ be the set of n multi-modal sensor nodes randomly deployed throughout Ω. Let the position of a target $\tau_\ell \in \mathcal{T}$ at time k be denoted as $\mathbf{u}^{\tau_\ell}(k) = (x^{\tau_\ell}, y^{\tau_\ell})(k) \in \Omega$, while that of a node s_i be denoted as $\mathbf{u}^{s_i} = (x^{s_i}, y^{s_i}) \in \Omega$.

2.1 Sensor Node Description

Definition 2.1.1 (Multi-modal Sensor Node) *A multi-modal sensor node, shown in Fig. 2.1, is an autonomous agent that contains a heterogeneous sensor suite, a data processing unit (DPU), a transmitter/receiver, and a Global Positioning System (GPS). The sensor suite contains several Low Power Sensing (LPS) devices, which are passive binary detectors consuming very little energy (e.g., Passive Infrared (PIR) sensors) with a fixed sensing range R_{LPS}. It also contains High Power Sensing (HPS) devices, which are active sensors providing the range and azimuth measurements of*

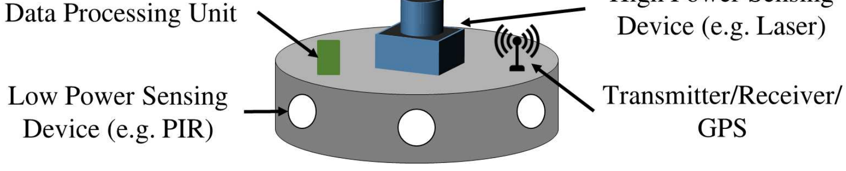

Data Processing Unit
High Power Sensing Device (e.g. Laser)
Low Power Sensing Device (e.g. PIR)
Transmitter/Receiver/ GPS

Figure 2.1: Example of a Multi-modal Sensor Node

targets (e.g., Laser Range Finders) [44] within a fixed sensing range R_{HPS}. The DPU facilitates decision-making to enable or disable each device at time k, while the transmitter/receiver are used to facilitate distributed collaboration.

Definition 2.1.2 (Sensor Node Coverage Area) *The coverage area of a sensor node $s_i \in \mathcal{S}$ at time k is defined as*

$$\Omega^{s_i}(k) \triangleq \{(x, y) \in \Omega : ||(x, y) - \mathbf{u}^{s_i}|| \leq R^{s_i}_{HPS}(k)\}. \tag{2.1}$$

and is the region where a target may be detected. This is based on the HPS device range at time k, where $R^{s_i}_{HPS}(k) = 0m$ if the HPS device is disabled and $R^{s_i}_{HPS}(k) = R_{HPS}$ if it is enabled.

Definition 2.1.3 (Sensor Node Neighborhood) *The neighborhood of a sensor node $s_i \in \mathcal{S}$ is defined as*

$$\mathcal{N}^{s_i} \quad \triangleq \quad \{s_j \in \{\mathcal{S} \setminus s_i\} : ||\mathbf{u}^{s_j} - \mathbf{u}^{s_i}|| \leq R_c\}, \tag{2.2}$$

where R_c is the communication radius of the node.

The total energy consumed [26] by a sensor node s_i until time k is computed as

$$E^{s_i}(k) \quad = \quad \sum_k \sum_j e^{s_i}_j . \chi^{s_i}_j(k) \Delta T, \tag{2.3}$$

where $e^{s_i}_j$ denotes the rate of energy consumption per unit time by a certain device $j \in \{$DPU, LPS, HPS, transmitter (TX), receiver (RX), clock$\}$; $\chi^{s_i}_j(k) \in \{0, 1\}$ indicates whether the device is *ON* or *OFF* at time k; and ΔT is the sample time interval. Thus, the total energy consumed by the entire sensor network until time k is given as $E_{net}(k) = \sum_{i=1}^{n} E^{s_i}(k)$.

2.2 Target Detection and Measurement Models

The motion of a target, τ_ℓ, is modeled using a *Discrete White Noise Acceleration* (DWNA) model [45] as follows

$$\mathbf{x}(k+1) \quad = \quad \mathbf{f}(\mathbf{x}(k), k) + \boldsymbol{\upsilon}(k) \tag{2.4}$$

where $\mathbf{x}(k) \triangleq [x(k), \dot{x}(k), y(k), \dot{y}(k), \psi(k)]'$ is the target state at time k, which includes the position and velocity in x and y and the turning rate $\psi(k)$; $\mathbf{f}(\mathbf{x}(k), k)$ is the state transition matrix, $\boldsymbol{\upsilon}(k)$ is the zero-mean white Gaussian process noise. The target is assumed to travel according to the nearly coordinated turning model. Details are in [45].

A sensor node s_i can use it's LPS devices to detect a target τ_ℓ. We adopt the detection model proposed in [46]:

$$P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k)) = \begin{cases} \alpha & ||\mathbf{u}^{\tau_\ell}(k) - \mathbf{u}^{s_i}|| < R_r \\ \alpha e^{-\beta(||\mathbf{u}^{\tau_\ell}(k) - \mathbf{u}^{s_i}|| - R_r)} & R_r \leq ||\mathbf{u}^{\tau_\ell}(k) - \mathbf{u}^{s_i}|| \leq R_{LPS} \\ p_{fa} & ||\mathbf{u}^{\tau_\ell}(k) - \mathbf{u}^{s_i}|| > R_{LPS} \end{cases} \tag{2.5}$$

where $P^{s_i}_{D,LPS}$ is the detection probability of sensor s_i; R_r is the reliable sensing radius of the LPS device; α is the detection probability within reliable range R_r; β is the decay rate of detection probability with distance; and $p_{fa} = 1 - e^{-f\Delta T}$ is the probability of a false alarm [47] with a false alarm rate f during a ΔT second scan.

A sensor node s_i can use it's HPS devices to collect the range and azimuth measurements $\mathbf{z}(k) = \{\mathbf{z}_1(k), ..., \mathbf{z}_o(k)\}$ of the target at time k. Each measurement is modeled as

$$\mathbf{z}_j(k) \quad = \quad \mathbf{h}(\mathbf{x}(k), k) + \mathbf{w}(k) \tag{2.6}$$

where $\mathbf{h}(\mathbf{x}(k), k)$ is a nonlinear measurement model [45] and $\mathbf{w}(k)$ is the zero-mean white Gaussian measurement noise. The measurements are received with a probability $P^{s_i}_{D,HPS} = \gamma$ if the target lies within the node's sensing range R_{HPS}. However, the measurements $\mathbf{z}(k)$ may also contain some false measurements along with the true target measurements. These false measurements, known as clutter, are generated according to a Poisson distribution with mean μ_{cl} [48].

2.3 Resilient and Energy-efficient Target Coverage Problem

We focus on the target coverage problem whose objective is to maximize the network lifetime while ensuring that each moving target τ_ℓ is covered by $N_{sel} \geq 1$ nodes using their HPS devices at all times to facilitate tracking.

To ensure energy-efficiency, our objective is to not cover the whole area but only the areas where the target is currently present and predicted to be during the next time step, thus enabling opportunistic sensing. However, the network may develop coverage gaps over time due to non-uniform node distribution, sensor failures, node drifting, etc., which can affect the tracking performance.

Definition 2.3.1 (Coverage Gap) *Given the coverage areas $\Omega^{s_i}(k)$, $\forall s_i \in \mathcal{S}$, based on their sensing ranges $R_{HPS}^{s_i}(k)$, a connected region $G(k) \subset \Omega$ is defined as a coverage gap if*

$$G(k) = \left\{ (x, y) \in \Omega : (x, y) \notin \bigcup_{s_i \in \mathcal{S}} \Omega^{s_i}(k) \right\}. \tag{2.7}$$

Definition 2.3.2 (Target Coverage) *Target coverage is said to be achieved at time k if $\mathbf{u}^{\tau_\ell}(k) \notin G(k), \forall \tau_\ell \in \mathcal{T}$, i.e. all targets are covered.*

Definition 2.3.3 (Coverage Degree) *Let $\mathcal{S}_c(\mathbf{u}^{\tau_\ell}(k)) = \{s_i \in \mathcal{S} : \mathbf{u}^{\tau_\ell}(k) \in \Omega^{s_i}(k)\}$ denote the set of all nodes that contain the target τ_ℓ within their coverage areas at time k. Then, the coverage degree of target τ_ℓ is defined as $D(\mathbf{u}^{\tau_\ell}(k)) = |\mathcal{S}_c(\mathbf{u}^{\tau_\ell}(k))|$.*

2.4 Thesis Objective

The objective of this thesis is to develop a network autonomy approach that utilizes distributed supervisors to probabilistically control multi-modal sensor nodes to meet the following requirements: (1) improved network life expectancy via opportunistic sensing, (2) resilient target coverage in the presence of coverage gaps, and (3) high tracking accuracy and low missed detection rates. The first two themes of this thesis focus on enabling opportunistic sensing into the network for energy efficiency. Whereas the third theme handles resilient target coverage in the presence of sensor failures and coverage gaps by enabling opportunistic coverage.

CHAPTER 3

PREDICTION-BASED OPPORTUNISTIC SENSING (POSE)

3.1 Energy-efficient Target Coverage Introduction

The *Target Coverage* problem consists of developing a sensor network that maximizes the network lifetime while ensuring that all targets are observed. Therefore, it is important that the sensor nodes utilize their difference devices in an energy-efficient manner to improve the network lifetime. A simple method is random scheduling which turns the devices on and off randomly to conserve energy [49]. Another simple approach is to schedule the nodes based on a fixed or adaptive duty-cycle. In contrast, there are other methods that optimize the activity of the network based on the local spatio-temporally varying demand [50]. These methods fall under the category of *Opportunistic Sensing* where the devices are enabled and disabled dynamically for efficient utilization of energy sources [51]. To perform opportunistic sensing in a pro-active manner, it is desired that the network has predictive intelligence to allow the sensor nodes to prepare for an event (e.g. target arrival) in advance.

For prediction-based opportunistic sensing, information must be communicated between sensor nodes; however, sharing information consumes energy [52], thus making it difficult to balance communication requirements and energy minimization. Specifically, in the multi-target tracking application, it is desired to track the targets with high power sensing devices opportunistically, i.e.

to turn on the HPS devices only on the nodes that are located in the region(s) where the target(s) are currently present or approaching. This requires accurate predictions of the targets' paths via distributed fusion of information received from multiple nodes detecting the targets'. However, the information received may be associated to different targets [53], corrupted by noise, or provide false beliefs of targets present in the environment [54]. These issues can drastically increase the tracking error if data validation, association, and filtering are not incorporated into the DSN.

This chapter addresses these issues by presenting a distributed supervisory control algorithm, called *Prediction-based Opportunistic Sensing* (POSE) [41]. This algorithm manages the power consumption of each multi-modal sensor node in a distributed fashion to enable energy-efficient target coverage while minimizing tracking error and missed detection rates. Each multi-modal node consists of multiple devices including HPS, LPS, DPU, TX, and RX, as discussed in Chapter 2.1. Each node also contains an embedded supervisor, i.e. a *Probabilistic Finite State Automaton* (PFSA), which enables and disables the devices on the node in an opportunistic manner to conserve energy.

The states of the PFSA represent the different activities of the node which correspond to enabling/disabling its various sensing and communication devices [8, 55]. The state transition probabilities of the PFSA are dynamically updated based on the information received from neighboring nodes or from direct observation of a target using the local sensor suite. These probabilities control the switching of the PFSA states to facilitate opportunistic sensing based on the predicted target(s) trajectories. For example, the PFSA of a sensor node enables its high power consuming devices when a target is present within its coverage region, while disabling them and entering into a low power consuming cycle when it is not present.

Consider a network shown in Figure 3.1, where the sensor nodes are randomly deployed and two targets are traveling through the ROI. Also, consider that the orange diamond nodes are the tracking or *HPS* state nodes, the green triangles are detection or *LPS* state nodes, and the white circles are sleeping nodes. The *HPS* state nodes are assumed to use their high power consuming devices to track targets while the *LPS* nodes use the low power devices for target detection and energy conservation. As seen in the figure, the sensor nodes form spatio-temporal clusters of *HPS*

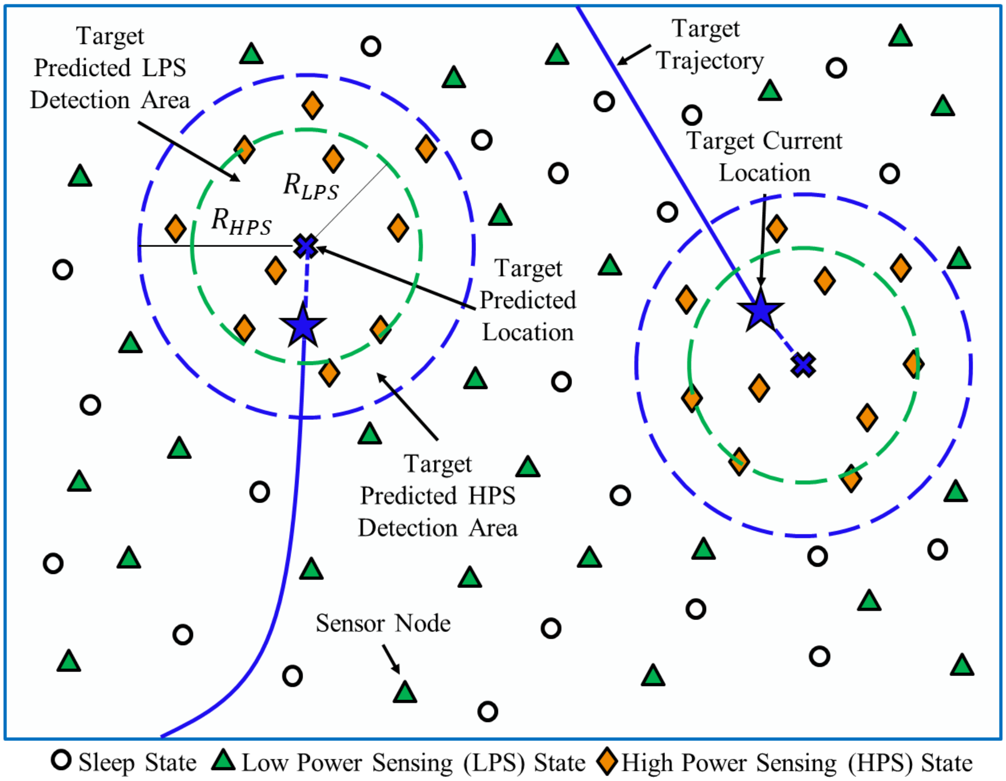

Target Predicted LPS Detection Area
Target Trajectory
R_{LPS}
R_{HPS}
Target Current Location
Target Predicted Location
Target Predicted HPS Detection Area
Sensor Node
○ Sleep State ▲ Low Power Sensing (LPS) State ◆ High Power Sensing (HPS) State

Figure 3.1: An Illustration of the Prediction-based Opportunistic Sensing Algorithm

state nodes in a distributed manner within the blue circle around the targets predicted position. Outside of the blue circle, the sensor nodes are conserving energy by staying in a low energy consuming state, i.e. *LPS* or *Sleep* state. The *HPS* state node clusters broadcast the target state information to alarm all of the *LPS* and *HPS* state nodes within the communication neighborhood. The *LPS* and *HPS* state nodes then use the received information to identify the target's location, predict the next position of the target, and accordingly transition to the *HPS* state during the next time step if the target is traveling into the node's HPS device coverage area. This prediction-based sensor activation method allows for the *HPS* state clusters to form around the target to track it with high accuracy in an opportunistic manner, while the remaining network conserves their energy.

The main contribution of this paper is the development of the POSE algorithm that is built upon distributed supervisors (i.e. PFSA) that enable scheduling of multi-modal sensor nodes consisting of several sensing devices that consume different amounts of energy. The POSE algorithm is validated by Monte Carlo simulations for multiple scenarios on a Matlab platform and compared with distributed scheduling schemes. The results show that the POSE algorithm provides significant energy savings and low missed detection rates while improving tracking accuracy via fusion-driven state initialization.

3.2 Related Work

Energy-efficient target tracking has been studied using various approaches throughout the DSN literature. To begin with, the sensor nodes must be deployed in a manner that allows for full coverage of the ROI [56, 57]. This can be achieved by optimally placing sensor nodes to maximize coverage [58, 59]; however this becomes impractical for large scale networks, in complex geographical regions, and due to the deployment time. To overcome this limitation, the κ-coverage problem was studied [37] to ensures that every point within the ROI is observable by at least κ sensor nodes at a given time. This assumes that the sensor nodes are deployed according to a random distribution, e.g. Uniform or Poisson [2]. Once the sensor nodes are deployed, energy-efficient sensor scheduling schemes are employed. In target tracking problems, the highest priority is accurate state estimation of the target. Therefore, adaptive sampling methods [60, 61] were developed to adjust the sampling interval to ensure accurate track estimation where the trace of the covariance matrix is used to represent the estimation accuracy.

Typical sensor scheduling approaches found in the literature are based on three types of network architectures: (1) Centralized, (2) Cluster-based, and (3) Distributed. Centralized networks require each sensor node to communicate their observations to a central computing station which calculates and transmits the optimal scheduling decisions to each sensor node while estimating the target trajectory [19, 62]. Some of these methods employ sensor selection techniques to minimize the number of sensors used for data collection [62, 63]. The centralized architectures can guarantee that the scheduling protocol will minimize the energy consumption of the network; however, these architectures are not scalable and thus very difficult to implement on large scale networks due to multiple factors such as communication limitations, transmission delays, and data processing requirements.

Cluster-based network architectures were proposed to overcome the limitations of centralized architectures by identifying a few sensor nodes throughout the ROI, to act as central computers, called *cluster heads* (CH) [30] [20]. Each CH forms an independent cluster which consists of multiple sensor nodes within its communication neighborhood. These sensor nodes send all the measurement data to their CH for analysis to generate state estimates for target tracking, sensor scheduling, and even for new cluster head selection [50, 61, 64]. The CH performs sensor scheduling by means of an optimization algorithm which selects the optimal subset of sensors within the cluster to track the target. This requires the knowledge of the positions of all the sensors within the cluster as well as their respective energy levels. Since the information measured outside of the cluster is unobservable, scheduling decisions in hybrid networks are only locally optimal.

The CH based methods allow for controlled energy consumption throughout the network to maximize its lifetime [65]. These methods eliminate the bandwidth limitation found in centralized networks by limiting a cluster's size to allow for single-hop communications. However, data transmission between the sensor nodes and the CH occur frequently [21], thus consuming high energy. Furthermore, if a CH were to fail, due to defects or energy depletion, the entire cluster would stop receiving scheduling decisions, rendering a large gap in the ROI [22], and requiring the implementation of dynamic CH election schemes [23].

Distributed network architectures have also been proposed where each sensor node dynamically controls its own tasks based on the information measured by its own sensor suite or perceived from its neighboring nodes [24]. The methods reported here deal with partial information broadcasting [66], adapting sensor frequency [67], and scheduling binary sensor nodes [26] for accurate target tracking with energy minimization. Unlike most cluster-based approaches, the distributed networks tend to be reactive in the sense that they only communicate information when an event has occurred, thus allowing for savings on transmission costs. Also, distributed architectures do not require sensor or CH selection, thus allowing for less computation requirements. Similar to cluster-based networks, the scheduling decisions here are only locally optimal due to limited observation region, however these architectures are scalable, provide fault tolerant attributes by construction, and eliminate the bandwidth limitations of centralized networks.

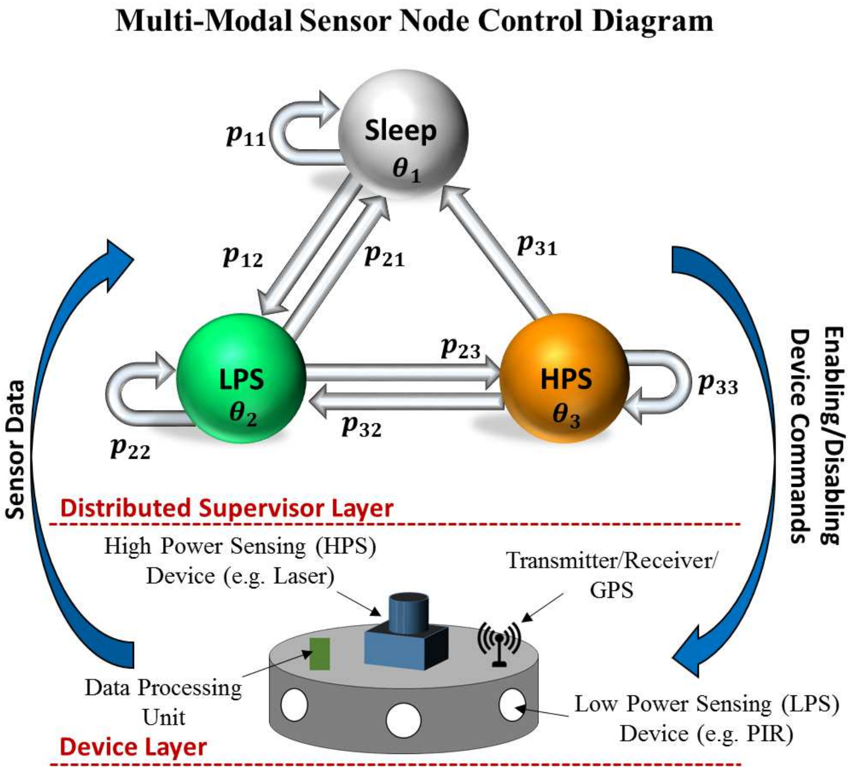

Multi-Modal Sensor Node Control Diagram
Sleep
θ_1
LPS
θ_2
HPS
θ_3
p_{11}
p_{12}
p_{21}
p_{31}
p_{23}
p_{32}
p_{22}
p_{33}
Sensor Data
Enabling/Disabling Device Commands
Distributed Supervisor Layer
High Power Sensing (HPS) Device (e.g. Laser)
Transmitter/Receiver/ GPS
Data Processing Unit
Low Power Sensing (LPS) Device (e.g. PIR)
Device Layer

Figure 3.2: Probabilistic Finite State Automaton (PFSA)-based supervisor embedded on each multi-modal sensor node for distributed scheduling

The POSE algorithm developed in this paper utilizes distributed supervisors that rely on target's state prediction to probabilistically control the scheduling of multi-modal sensor nodes for energy minimization. This algorithm provides significant energy savings that extend the network's lifetime extension while ensuring low missed detection rates and low tracking errors.

3.3 Distributed Supervisor: Probabilistic Finite State Automaton Overview

The scheduling of each multi-modal sensor node is controlled by a distributed PFSA-based supervisor, as shown in Fig. 3.2, which controls the devices enabled and disabled based on the information available to the sensor node.

Definition 3.3.1 (PFSA) *A PFSA [68] is defined as a* 3*-tuple* $\Xi = \langle \Theta, A, P \rangle$*, where*

- Θ *is a finite set of states,*
- A *is a finite alphabet,*

Table 3.1: Devices Enabled(✓)/Disabled(✗) at PFSA States.

States $\setminus$ Devices	DPU	LPS	HPS	Transmitter	Receiver
Sleep (θ_1)	✓	✗	✗	✗	✗
LPS (θ_3)	✓	✓	✗	✓	✓
HPS (θ_4)	✓	✗	✓	✓	✓

- $p : \Theta \times \Theta \to [0, 1]$ *are the state transition probabilities which form a stochastic matrix* $P \equiv [p_{ij}]$, *where* $p_{ij} \equiv p(\theta_i, \theta_j)$, $\forall \theta_i, \theta_j \in \Theta$, *s.t.* $\sum_{\theta' \in \Theta} p(\theta, \theta') = 1, \ \forall \theta \in \Theta$.

The alphabet is defined as $A = \{\epsilon, 0, 1\}$, where ϵ is the null symbol emitted when no information is available, 0 indicates no target detection, and 1 indicates target detection.

The state set Θ consists of three states: (1) *Sleep* (θ_1), (2) *LPS* (θ_2), and (3) *HPS* (θ_3), as shown in Fig. 3.2. These states have the following attributes:

- *Sleep* (θ_1): Disables all devices on the node to consume minimal energy,

- *LPS* (θ_2): Detects the target while conserving energy using either (i) LPS devices and/or (ii) information perceived by neighboring sensor nodes, and

- *HPS* (θ_3): Detects the target using the HPS devices, estimates its current state, alerts neighbors of its whereabouts, and predicts its next position to enable prediction-based opportunistic sensing.

Table 3.1 shows the various devices that are enabled or disabled for each PFSA state.

Consider a sensor node $s_i \in \mathcal{S}$ which can operate in one of the three states at one time. The PFSA-based supervisor runs a unique algorithm within each state to dynamically update it's state transition probabilities based on the information acquired about the targets' whereabouts. These probabilities control the transition of the node from one state to another. The algorithms within each state are described in the following subsections.

3.3.1 Sleep State Description

The *Sleep* state, θ_1, is designed to minimize energy consumption by disabling all devices on the node when no target is detected nearby. In the *Sleep* state, the sensor node either stays in the *Sleep* state with a probability $p_{11}^{s_i}(k) = p_{sleep}$ or makes a transition to the *LPS* state with

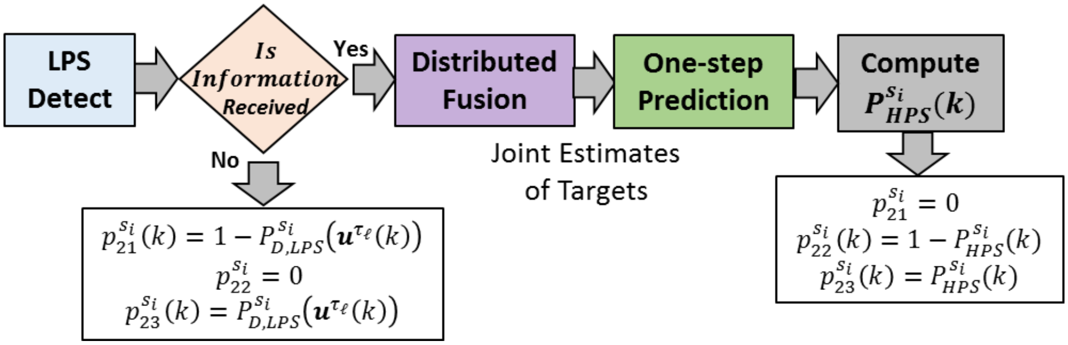

LPS Detect
Is Information Received
Yes
No
Distributed Fusion
One-step Prediction
Compute $P^{s_i}_{HPS}(k)$
Joint Estimates of Targets
$p^{s_i}_{21}(k) = 1 - P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$
$p^{s_i}_{22} = 0$
$p^{s_i}_{23}(k) = P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$
$p^{s_i}_{21} = 0$
$p^{s_i}_{22}(k) = 1 - P^{s_i}_{HPS}(k)$
$p^{s_i}_{23}(k) = P^{s_i}_{HPS}(k)$

Figure 3.3: *LPS* State Algorithm

a probability $p^{s_i}_{12}(k) = 1 - p_{sleep}$, where p_{sleep} is a design parameter that governs if the node will remain in the *Sleep* state. At the *Sleep* state θ_1, the following state transitions are possible: $\delta_{\theta_1} = \{(\theta_1, \epsilon, \theta_1), (\theta_1, \epsilon, \theta_2)\}$, where the null symbol ϵ denotes no observation of the target.

3.3.2 Low Power Sensing (LPS) State Description

The *LPS* state θ_2 is designed to conserve energy while enabling target detection. In this state, the LPS devices, the DPU, TX, and RX are powered on. Fig. 3.3 shows the flowchart of the algorithm within the *LPS* state.

• ***Target Detection***: Here a target could be detected either by: (i) using the LPS devices or by (ii) fusing the information received from neighbors. If a target, τ_ℓ, is present, then the sensor node s_i detects it with a probability $P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$ as per Eq. (2.5). Then, it checks if it has received any information from its neighbors. Let $\mathcal{N}^{s_i}_{HPS} \subseteq \mathcal{N}^{s_i}$ be the set of *HPS* state sensors that have broadcasted target state information to s_i. If $\mathcal{N}^{s_i}_{HPS} = \emptyset$, then none of the node's neighbors have detected and tracked a target located in the nodes local neighborhood. Therefore, node s_i relies on its own sensor information to govern its control action. If the LPS devices detect the target, then the node will transition to the *HPS* state with a probability $p_{23} = P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$. Whereas if a target is not detected, s_i will transition to the *Sleep* state with a probability $p_{21} = 1 - P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$. In either case, s_i will not remain in the *LPS* state because it has directly observed if a target is present or not. However, if information is received, i.e. $\mathcal{N}^{s_i}_{HPS} \neq \emptyset$, then the node performs the Distributed Fusion algorithm described next.

• ***Distributed Fusion***: The distributed fusion algorithm, shown in Fig. 3.4, fuses the received

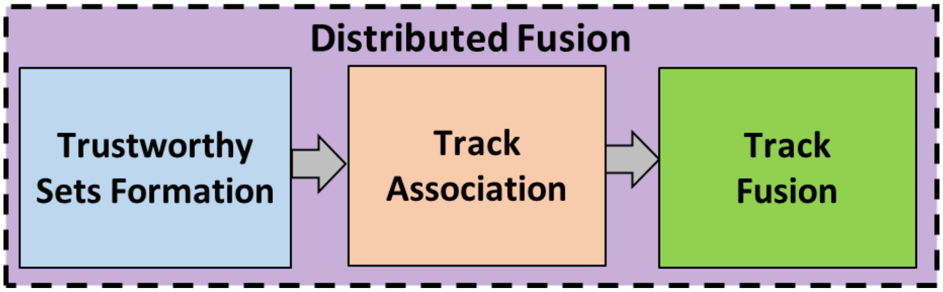


Figure 3.4: Distributed Fusion Algorithm

information to obtain joint estimates of the targets located within its communication neighborhood.

Consider a neighboring sensor node $s_j \in \mathcal{N}^{s_i}$ and consider that a target is present within its HPS sensing radius R_{HPS}. If this node is in the *HPS* state at time k, then it observes the target and generates estimates of the target state $\hat{\mathbf{x}}^{s_j}(k|k)$, its covariance $\hat{\mathbf{\Sigma}}^{s_j}(k|k)$, and the filter gain matrix $\hat{\mathbf{W}}^{s_j}(k)$ (For details see Section 3.3.3). Then, it broadcasts the information packet $(\hat{\mathbf{x}}^{s_j}(k|k), \hat{\mathbf{\Sigma}}^{s_j}(k|k), \hat{\mathbf{W}}^{s_j}(k))$ to its neighbors (assuming a single-hop network) within the communication radius, R_c. Since node s_i has its receiver enabled, it receives these packets with a probability $P_{RX}^{s_i} = 1 - p_{drop}$. A realistic communication model based on channel properties will be considered in future work. Let $\mathcal{N}_{HPS}^{s_i} \subseteq \mathcal{N}^{s_i}$ be the set of neighbors in the *HPS* state from which packets are successfully received. Then the following information ensemble is constructed:

$$\hat{\mathbf{I}}^{s_i}(k) = \left\{ \left(\hat{\mathbf{x}}^{s_j}(k|k), \hat{\mathbf{\Sigma}}^{s_j}(k|k), \hat{\mathbf{W}}^{s_j}(k) \right), \forall s_j \in \mathcal{N}_{HPS}^{s_i} \right\}, \tag{3.1}$$

This information is then combined using the following steps.

1) Trustworthy Set Formation: Since some of the received estimates might be poor due to noise and other factors, the information received must first be validated. This is done by forming a set of trustworthy neighbors $\mathcal{N}_T^{s_i} \subseteq \mathcal{N}_{HPS}^{s_i}$ by evaluating the sum of the position error as follows

$$\mathcal{N}_T^{s_i} = \{ s_j \in \mathcal{N}_{HPS}^{s_i} : Trace(\mathbf{H}(k)\hat{\mathbf{\Sigma}}^{s_j}(k|k)\mathbf{H}(k)') \leq \xi \} \tag{3.2}$$

where $\mathbf{H}(k)$ is the Jacobian of the measurement model defined in Section 2.2 and ξ is the maximum tolerance of the estimate. In this thesis, $\xi = \frac{R_{HPS}^2 \sigma_\phi^2 + \sigma_R^2}{2}$, where σ_ϕ and σ_R are the standard deviations in the azimuth and range measurements of the HPS sensor. This tolerance is dervived from the state initialization method used in this work. If the position covariance error increases above the initialized value, then the estimation error is not decreasing and may be a track generated by false measurements. Finally, node s_i receives the following trustworthy information:

$$\hat{\mathbf{I}}_T^{s_i}(k) = \big\{\big(\hat{\mathbf{x}}^{s_j}(k|k), \hat{\boldsymbol{\Sigma}}^{s_j}(k|k), \hat{\mathbf{W}}^{s_j}(k)\big), \forall s_j \in \mathcal{N}_T^{s_i}\big\}, \tag{3.3}$$

Remark 3.3.1 *Forming the Trustworthy Information Ensemble,* $\hat{\mathbf{I}}_T^{s_i}(k)$ *allows for the following advantages:*

i False tracks and poor state estimates are eliminated,

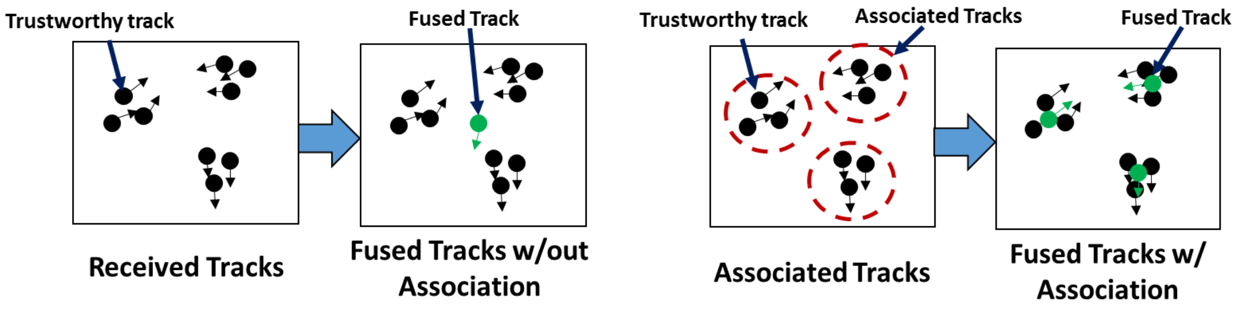
ii State estimation and fusion are enhanced by combining accurate estimates, and

iii the computational complexity is reduced by filtering out unnecessary date.

2) Track Association: Since there could be multiple targets within a close proximity of eachother, the trustworthy information, $\hat{\mathbf{I}}_T^{s_i}(k)$, may contain information related to different targets. Therefore, the estimates must be associated together to improve state vector fusion, while allowing node s_i to understand how many targets are located within its local neighborhood. Since each sensor node in the network uses the same process noise in the motion model [69], each estimate received is correlated and the traditional Mahalanobis Distance Metric cannot be used. To overcome this, Bar-shalom [69] proposed the *Track-to-Track Association* (T2TA) algorithm which associates the tracks using the following metric

$$\begin{aligned} \varrho_{jh}^2 \quad = \quad & \big(\hat{\mathbf{x}}^{s_j}(k|k) - \hat{\mathbf{x}}^{s_h}(k|k)\big)' \big[\hat{\boldsymbol{\Sigma}}^{s_j}(k|k) + \hat{\boldsymbol{\Sigma}}^{s_h}(k|k) - \hat{\boldsymbol{\Sigma}}^{s_j,s_h}(k|k) - \hat{\boldsymbol{\Sigma}}^{s_h,s_j}(k|k)\big]^{-1} \\ & \cdot \big(\hat{\mathbf{x}}^{s_j}(k|k) - \hat{\mathbf{x}}^{s_h}(k|k)\big) \end{aligned} \tag{3.4}$$

where ϱ_{jh}^2 is the association metric between sensors s_j and s_h which belong to the set of trustworthy neighbors $\mathcal{N}_T^{s_i}$; and $\big(\hat{\mathbf{x}}^{s_j}(k|k),\, \hat{\boldsymbol{\Sigma}}^{s_j}(k|k)\big) \in \hat{\mathbf{I}}_T^{s_i}(k)$ and $\big(\hat{\mathbf{x}}^{s_h}(k|k),\, \hat{\boldsymbol{\Sigma}}^{s_h}(k|k)\big) \in \hat{\mathbf{I}}_T^{s_i}(k)$ are the state and covariance estimates for a target at time k, for sensors s_j and s_h respectively. The cross covariance term $\hat{\boldsymbol{\Sigma}}^{s_j,s_h}(k|k)$ in Eq. (3.4), between sensor s_j and s_h is computed as follows.



(a) Example of fusion without association

(b) Eample of fusion with association

Figure 3.5: Benefits of state association

$$\begin{aligned}\hat{\mathbf{\Sigma}}^{s_j,s_h}(k|k) &= \big[\mathbf{I}-\hat{\mathbf{W}}^{s_j}(k)\cdot\mathbf{H}(k)\big]\mathbf{F}(k)\cdot\hat{\mathbf{\Sigma}}^{s_j,s_h}(k-1|k-1)\cdot\mathbf{F}(k)'\big[\mathbf{I}-\hat{\mathbf{W}}^{s_h}(k)\cdot\mathbf{H}(k)\big]' \\ &\quad +\big[\mathbf{I}-\hat{\mathbf{W}}^{s_j}(k)\cdot\mathbf{H}(k)\big]\cdot\mathbf{Q}\cdot\big[\mathbf{I}-\hat{\mathbf{W}}^{s_h}(k)\cdot\mathbf{H}(k)\big]' \qquad (3.5)\end{aligned}$$

where $\hat{\mathbf{W}}^{s_j}(k) \in \hat{\mathbf{I}}_T^{s_i}(k)$; $\hat{\mathbf{W}}^{s_h}(k) \in \hat{\mathbf{I}}_T^{s_i}(k)$; $\mathbf{I}$ is an identity matrix; $\mathbf{H}(k)$ is the Jacobian of the measurement model; $\mathbf{F}(k)$ is the Jacobian of the state transition matrix; $\mathbf{Q}$ is the process noise covariance matrix; and $\hat{\mathbf{\Sigma}}^{s_i,s_j}(k-1|k-1)$ is the previous cross covariance between sensors s_j and s_h.

To implement T2TA, the sensor node s_i forms groups by associating the estimates received from its trustworthy neighbors using the χ^2-Test [69], as shown in Fig. 3.5b on the left. This is done in a manner such that any two sensors, say s_j and s_h, within a group, must satisfy $\varrho_{jh}^2 \leq \chi_d^2(\gamma)$, where $d = 2$ is the number of measurements and $\gamma = 0.05$ is the level of significance.

Thus each sensor from the set of trustworthy neighbors is assigned to one of the groups $\mathcal{N}_T^{s_i,c}$, $c = 1, 2, ...C$, such that

$$\bigcup_{c=1}^{C} \mathcal{N}_T^{s_i,c} = \mathcal{N}_T^{s_i} \qquad (3.6)$$

and the intersection between any two groups is empty. An example of 3 groups of associated trustworthy tracks are shown in Fig. 3.5b. Note that C may not necessarily be equal to the total number of targets m in the ROI. This is because not all targets may have been observed by the same group of sensors.

The corresponding estimates within the trustworthy information are grouped to form the information ensembles, $\hat{\mathbf{I}}_T^{s_i,c}(k) \subseteq \hat{\mathbf{I}}_T^{s_i}(k), \forall c = 1, ...C$.

This procedure is a very important step for each sensor node since it needs to develop a reliable state estimate corresponding to every target that is in its neighborhood. If the sensor node were to bypass this association, the fused state estimate would be inaccurate and lead to poor control strategies. For example, consider the trustworthy state estimates shown in Fig. 3.5a on the left. If the association step is not implemented, all of the state estimates would be fused together to form the single estimate shown in Fig. 3.5a on the right. Whereas if association is included, as shown in Fig. 3.5b, the resulting fused state estimates represent the number of disjoint tracks present and an accurate representation of the true target locations.

3) Track Fusion: Once the information received from neighbors is filtered and associated to form C trustworthy information ensembles, $\hat{\mathbf{I}}_T^{s_i,c}(k) \subseteq \hat{\mathbf{I}}_T^{s_i}(k), \forall c = 1, ...C$, the next step is to fuse the associated information to generate a total of C target state estimates, as seen in Fig. 3.5b on the right. This paper uses the *Track-to-Track Fusion* algorithm [70], which combines multiple state estimates into a single estimate.

For fusion, the following procedure is performed. For each group $c = 1, 2, ...C$, the cross covariance matrix $\widetilde{\mathbf{\Sigma}}^{s_i,c}(k)$ between all $s_j, s_h \in \mathcal{N}_T^{s_i,c}$ is constructed as follows

$$\widetilde{\mathbf{\Sigma}}^{s_i,c}(k) = \begin{bmatrix} \vdots & \vdots & \vdots \\ \cdots & \hat{\mathbf{\Sigma}}^{s_j,s_h}(k|k) & \cdots \\ \vdots & \vdots & \vdots \end{bmatrix}, where \quad \hat{\mathbf{\Sigma}}^{s_j,s_h}(k|k) = \begin{cases} \hat{\mathbf{\Sigma}}^{s_j}(k|k) & \text{if } s_j = s_h \\ \hat{\mathbf{\Sigma}}^{s_j,s_h}(k|k) & \text{otherwise} \end{cases}, \tag{3.7}$$

$\hat{\mathbf{\Sigma}}^{s_j}(k|k) \in \hat{\mathbf{I}}_T^{s_i,c}(k)$, and $\hat{\mathbf{\Sigma}}^{s_j,s_h}(k|k)$ is given in Eq. 3.5. Then, the node s_i computes the fused state and covariance estimates from its trustworthy associated groups as follows

$$\hat{\mathbf{\Sigma}}^{s_i,c}(k|k) \quad = \quad \left(\widetilde{\mathbf{I}}(\widetilde{\mathbf{\Sigma}}^{s_i,c}(k))^{-1}\widetilde{\mathbf{I}}'\right)^{-1} \tag{3.8}$$

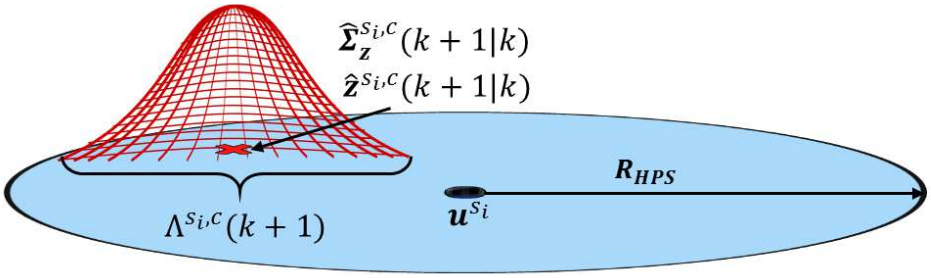


Figure 3.6: Visual representation of the probability $P^{s_i}_{HPS}(k)$

$$\hat{\mathbf{x}}^{s_i,c}(k|k) \quad = \quad \hat{\boldsymbol{\Sigma}}^{s_i,c}(k|k)\widetilde{\mathbf{I}}(\widetilde{\boldsymbol{\Sigma}}^{s_i,c}(k))^{-1}\begin{bmatrix} \vdots \\ \hat{\mathbf{x}}^{s_j}(k|k) \\ \vdots \end{bmatrix} \tag{3.9}$$

where $\widetilde{\mathbf{I}} = \begin{bmatrix} \mathbf{I}...\mathbf{I} \end{bmatrix}$ consists of $|\mathcal{N}_T^{s_i,c}|$ $n \times n$ identity matrices and $\hat{\mathbf{x}}^{s_j}(k|k) \in \hat{\mathbf{I}}_T^{s_i,c}(k)$. In this manner single estimates are produced for each associated group in the trustworthy set.

• ***One-Step Prediction***: Once all the received state information is fused, node s_i computes a one-step prediction [45] of the target's state using the Extended Kalman Filter:

$$\begin{aligned} \hat{\mathbf{x}}^{s_i,c}(k+1|k) \quad &= \quad \mathbf{f}(\hat{\mathbf{x}}^{s_i,c}(k|k), k) \\ \hat{\boldsymbol{\Sigma}}^{s_i,c}(k+1|k) \quad &= \quad \mathbf{F}(k)\hat{\boldsymbol{\Sigma}}^{s_i,c}(k|k)\mathbf{F}(k)' + \mathbf{Q}, \end{aligned} \tag{3.10}$$

where $\mathbf{F}(k)$ is the Jacobian of the state transition matrix evaluated at $\hat{\mathbf{x}}^{s_i,c}(k|k)$ and $\mathbf{Q}$ is the process noise covariance. This predicted state is used to computed the probability that the target is traveling within the node's coverage area, $\Omega^{s_i}(k)$.

• ***Computation of*** $P^{s_i}_{HPS}(k)$: If the target is predicted the travel into the coverage area of node s_i, i.e. $\Omega^{s_i}(k)$, then, node s_i should transition to the *HPS* state to track the target. However, if the target is not traveling within $\Omega^{s_i}(k)$, the node should not waste its energy by transitioning to the *HPS* state. This is achieve by using the probability of detecting the target with the HPS device, $P^{s_i}_{HPS}(k)$, as shown in Fig. 3.6, to influence the node's state transition and enable opportunistic sensing.

Therefore, once the one-step predictions are obtained for the C tracks, the sensor node s_i computes the probability of each target traveling through its HPS device coverage area at time $k+1$, as follows:

$$\Lambda^{s_i,c}(k+1) = \int\int_{\Omega^{s_i}(k+1)} N\Big(\hat{\mathbf{z}}^{s_i,c}(k+1|k), \hat{\mathbf{\Sigma}}_{\mathbf{z}}^{s_i,c}(k+1|k)\Big)dxdy \tag{3.11}$$

where $\Omega^{s_i}(k+1)$ is the nodes coverage area, $\hat{\mathbf{z}}^{s_i,c}(k+1|k) = \mathbf{h}(\hat{\mathbf{x}}^{s_i,c}(k+1|k), k)$ is the predicted location of the target, and $\hat{\mathbf{\Sigma}}_{\mathbf{z}}^{s_i,c}(k+1|k) = \mathbf{H}(k)\hat{\mathbf{\Sigma}}^{s_i,c}(k+1|k)\mathbf{H}(k)'$ is the predicted position covariance error of the target. Then the maximum probability of target detection over all tracks is

$$P_{HPS}^{s_i}(k) = \max_{c}\left\{\Lambda^{s_i,c}(k+1)\right\}, \tag{3.12}$$

which it uses to update the node's state transition probabilities.

• ***Updating the State Transition Probabilities***: Finally, node s_i updates it's PFSA probabilities as follows:

- *if* ($\mathcal{N}_{HPS}^{s_i} = \emptyset$), *i.e. Information was not received by node* s_i, *then*:
  $p_{21}^{s_i}(k) = 1 - P_{D,LPS}^{s_i}(\mathbf{u}^{\tau_\ell}(k))$; $p_{22}^{s_i}(k) = 0$; $p_{23}^{s_i}(k) = P_{D,LPS}^{s_i}(\mathbf{u}^{\tau_\ell}(k))$

- *if* ($\mathcal{N}_{HPS}^{s_i} \neq \emptyset$), *i.e. Information was received by node* s_i, *then*:
  $p_{21}^{s_i}(k) = 0$; $p_{22}^{s_i}(k) = 1 - P_{HPS}^{s_i}(k)$; $p_{23}^{s_i}(k) = P_{HPS}^{s_i}(k)$

To summarize, if information is not received by node s_i, the node will transition to the *HPS* state if a target is detected with the LPS devices. If a target is not detected, then it will transition to the *Sleep* state. The node will not remain in the *LPS* state because it has already observed that a target is not located in it's local neighborhood. Furthermore, it's neighbors have not transmitted information, thus indicating to node s_i that a target is not located within it's communication neighborhood. Therefore, it would be a waste of energy to continue in the *LPS* state.

However, if information is received from it's neighbors, the node should remain aware in the *LPS* or *HPS* state and not transition to the *Sleep* state. This is because the target is located within it's communication neighborhood and could be traveling through it's coverage area in the future. If the target is predicted to travel within it's HPS device coverage area during the next time step,

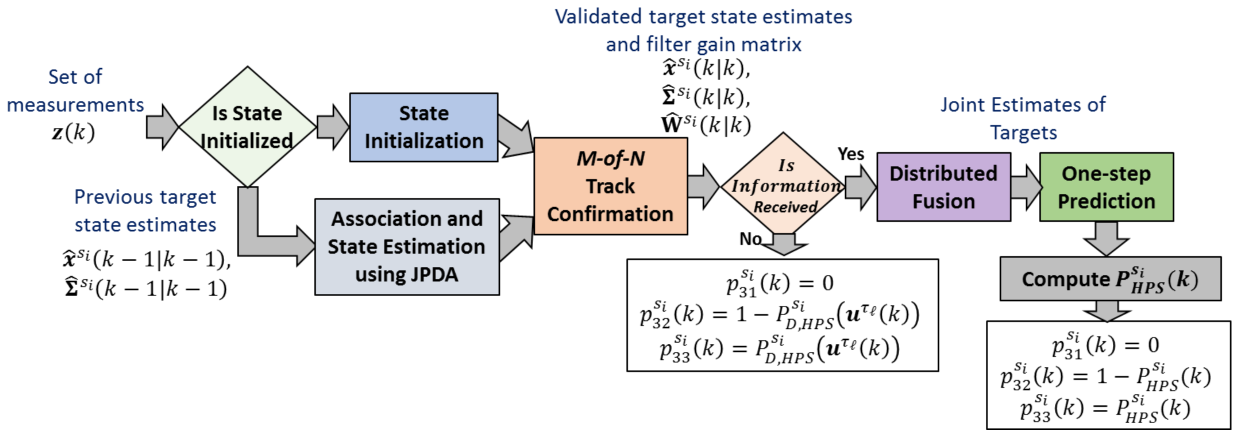

Validated target state estimates and filter gain matrix
$\hat{\boldsymbol{x}}^{s_i}(k|k)$,
$\hat{\boldsymbol{\Sigma}}^{s_i}(k|k)$,
$\hat{\mathbf{W}}^{s_i}(k|k)$
Set of measurements $\boldsymbol{z}(k)$
Is State Initialized
State Initialization
M-of-N Track Confirmation
Is Information Received
Yes
Distributed Fusion
Joint Estimates of Targets
One-step Prediction
Previous target state estimates
$\hat{\boldsymbol{x}}^{s_i}(k-1|k-1)$,
$\hat{\boldsymbol{\Sigma}}^{s_i}(k-1|k-1)$
Association and State Estimation using JPDA
No
$p_{31}^{s_i}(k) = 0$
$p_{32}^{s_i}(k) = 1 - P_{D,HPS}^{s_i}(\boldsymbol{u}^{\tau_\ell}(k))$
$p_{33}^{s_i}(k) = P_{D,HPS}^{s_i}(\boldsymbol{u}^{\tau_\ell}(k))$
Compute $\boldsymbol{P}_{HPS}^{s_i}(\boldsymbol{k})$
$p_{31}^{s_i}(k) = 0$
$p_{32}^{s_i}(k) = 1 - P_{HPS}^{s_i}(k)$
$p_{33}^{s_i}(k) = P_{HPS}^{s_i}(k)$

Figure 3.7: High Power Sensing state algorithm

i.e. $P_{HPS}^{s_i}(k)$ is close to 1, the node should transition to the *HPS* state to track the target. However, if $P_{HPS}^{s_i}(k)$ is low, then the node should remain aware in the *LPS* state in case the target travels in it's coverage area during a future time step.

3.3.3 High Power Sensing (HPS) State Description

The *HPS* state, θ_3, is designed to estimate the state of the target and track it when it is located within its sensing radius R_{HPS}. Here, the HPS device, DPU, TX, and RX are enabled and the LPS device is disabled. In this state, the sensor node will use the measurements obtained by the HPS devices to update it's target state estimates. Then, the node validates the updated tracks to ensure that it is not a false track. The validated target state $\hat{\mathbf{x}}^{s_i}(k|k)$, covariance $\hat{\mathbf{\Sigma}}^{s_i}(k|k)$, and filter gain matrix $\hat{\mathbf{W}}^{s_i}(k)$ are then broadcasted to the nodes within its neighborhood. Since this state uses more energy than any other state of the PFSA, it should only be enabled when the target is predicted to travel within its sensing radius R_{HPS}. Fig. 3.7 shows the various steps of the *HPS* state.

• ***Data Association and State Estimation***: In the *HPS* state, node s_i first receives a set of measurements, $\mathbf{z}(k)$, from its HPS devices. Subsequently, the previous state estimates $\hat{\mathbf{x}}^{s_i}(k-1|k-1)$, $\hat{\mathbf{\Sigma}}^{s_i}(k-1|k-1)$ are updated using the *Joint Probabilistic Data Association* (JPDA) filter [48] to generate $\hat{\mathbf{x}}^{s_i}(k|k)$, $\hat{\mathbf{\Sigma}}^{s_i}(k|k)$. For details on the JPDA algorithm, please see Appendix B. If the received measurements do not associate to the previous state estimate (e.g., when the target

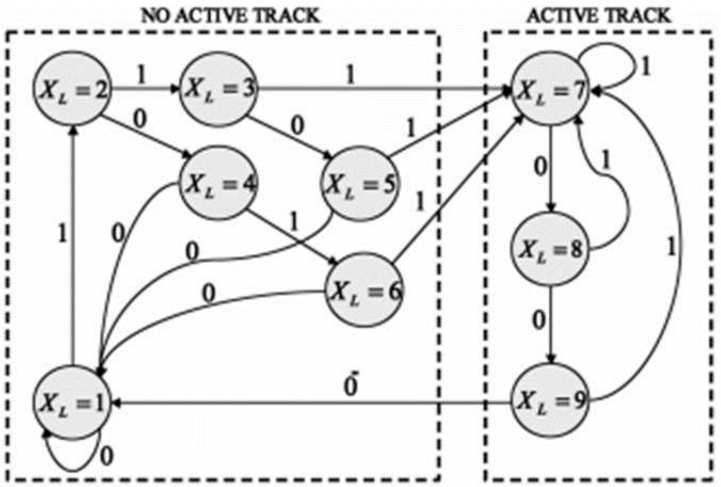


Figure 3.8: Markov chain model for track validation [72]

state is not initialized), then the node s_i must first perform state initialization. An example of the state initialization method used in the thesis is the *Automatic Track Formation* method [71] (described in Appendix A). However, other initialization method could be supplemented.

The JPDA filter was chosen in this thesis because it allows the sensor node, s_i, to update multiple state estimates at once. If multiple targets are located within $\Omega^{s_i}(k)$, then the JPDA will identify the measurements that relate to each target using measurement association. Therefore, the state estimates are only updated with the associated measurements, thus improving state estimation in the presence of clutter. It should also be noted that if the user prefers a different state estimation algorithm, the JPDA block in the *HPS* state algorithm (shown in Fig. 3.7) can be substituted with their preferred method.

In addition to the target range and azimuth measurements, $\mathbf{z}(k)$ may contain false alarms due to clutter, which can generate false tracks at each node. To ensure that false tracks do not propagate throughout the network, node s_i utilizes the M-of-N *Track Confirmation Logic* [72] to allow the network to be robust to false alarms. This approach has the node s_i maintain a Markov chain model, as shown in Fig. 3.8, that governs if the the state estimate is an active track or not. This approach ensures that M out of N consecutive time steps have a measurement associated to a target state estimate before the node confirms that it is an active track. Furthermore, once the target track has been confirmed, the node can only drop the track if M consecutive time steps so not have a measurement associate to it. However, if $< M$ out of N time steps have measurements associate to the target, the track is considered a false track and dropped from consideration. Once a target state estimate is classified as a valid track, node s_i will broadcast the target state $\hat{\mathbf{x}}^{s_i}(k|k)$, covariance $\hat{\mathbf{\Sigma}}^{s_i}(k|k)$, and filter gain matrix $\hat{\mathbf{W}}^{s_i}(k)$ to the other sensor nodes within its neighborhood.

• ***Distributed Fusion***: Since s_i is in the *HPS* state and has broadcasted information to it's neighbors, we define $\mathcal{N}^{s_i}_{HPS} = \mathcal{N}^{s_i}_{HPS} \cup \{s_i\}$. However, if s_i has not confirmed a target track, then $\mathcal{N}^{s_i}_{HPS}$ does not include s_i. Then, if $\mathcal{N}^{s_i}_{HPS} \neq \emptyset$, it will run the distributed fusion algorithm (please see Section 3.3.2 for details) in the same manner as in the *LPS* state. This generates the fused state estimates, $\hat{\mathbf{x}}^{s_i,c}(k|k)$ and $\hat{\mathbf{\Sigma}}^{s_i,c}(k|k)$, that allow the node to understand the location of targets within their communication neighborhood.

• ***One-Step Prediction***: Next, the node computes the predicted target state estimates according to Equation 3.10 in Section 3.3.2.

• ***Computation of*** $P^{s_i}_{HPS}(k)$: Then, node s_i computes the probability of the target traveling within s_i's coverage area $\Omega^{s_i}(k)$ during the next time step using Equation 3.12 in Section 3.3.2.

• ***Updating the State Transition Probabilities***: Finally, node s_i updates it's PFSA probabilities as follows:

- *if* ($\mathcal{N}^{s_i}_{HPS} = \emptyset$), s_i *has not received target information and has not validated a track, then*:
  $p^{s_i}_{31}(k) = 0$; $p^{s_i}_{32}(k) = 1 - P^{s_i}_{D,HPS}(\mathbf{u}^{\tau_\ell}(k))$; $p^{s_i}_{33}(k) = P^{s_i}_{D,HPS}(\mathbf{u}^{\tau_\ell}(k))$
- *if* ($\mathcal{N}^{s_i}_{HPS} \neq \emptyset$), s_i *has validated a track and/or has received target information from neighbors, then*:
  $p^{s_i}_{31}(k) = 0$; $p^{s_i}_{32}(k) = 1 - P^{s_i}_{HPS}(k)$; $p^{s_i}_{33}(k) = P^{s_i}_{HPS}(k)$

In summary, information is not receive by node s_i if its neighbors have not validated that a target is located within their coverage area, and node s_i has not validated a target track yet. In this condition, the node is desired to stay in the *HPS* state with the detection probability $P^{s_i}_{D,HPS}(\mathbf{u}^{\tau_\ell}(k))$ if the HPS devices have provided measurements of a target. This is because node s_i might have a target state initialized, but the state may not be considered a valid track yet. Therefore, the node should remain tracking the uncertain target. However, if a target is not located within its coverage area and information is not received, then the node should transition to the *LPS* state. This is because node s_i does not want to waste energy, but wants to be aware of its communication neighborhood since a target detection drove the node to be in the *HPS* state during the previous time step.

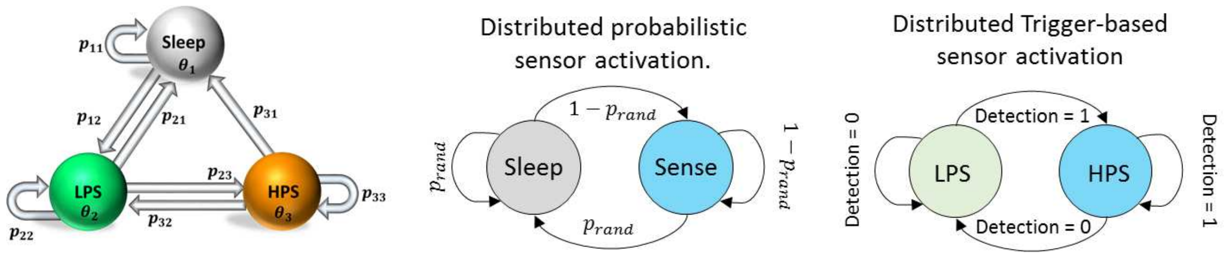
Alternatively, if node s_i receives information from it's neighbors, distributed fusion, one-step prediction, and computation of $P^{s_i}_{HPS}(k)$ has occurred. Therefore, the node should transition to the *HPS* state if the probability of the target traveling through its coverage area, $P^{s_i}_{HPS}(k)$, is high. Whereas if $P^{s_i}_{HPS}(k)$ is low, the node should transition to the *LPS* state to stay aware of its communication neighborhood. In both conditions, the node should not transition to the *Sleep* state because a target has been detected within the nodes communication neighborhood. Therefore, the target could travel toward the node in future time steps, requiring the node to maintain awareness in the *LPS* state.

3.4 POSE Algorithm Performance

To validate that the first theme of this thesis achieves energy-efficient target coverage, the POSE algorithm is compared against existing distributed scheduling methods. The POSE algorithm was simulated for 500 Monte Carlo runs in a $500m \times 500m$ Matlab environment with a single target traveling through the region. During each Monte Carlo run, the location of each sensor node is regenerated according to a uniform distribution. Each multi-modal sensor node is assumed to possess Passive Infrared (PIR) sensors as the LPS devices and a laser range finder as the HPS devices. The simulation energy costs, process noises ($\sigma_{\boldsymbol{v},x}, \sigma_{\boldsymbol{v},y}, \sigma_{\boldsymbol{v},\psi}$), measurement noises ($\sigma_R, \sigma_\phi$), and sensing range parameters are presented in Table 5.1.

Table 3.2: Simulation Parameters

$e_{clock} = 0.01W$	$\Delta T = 0.5s$	$\alpha = 0.95$
$e_{LPS} = 115mW$	$R_r = 15m$	$\beta = 0.0036$
$e_{HPS} = 12W$	$R_{LPS} = 30m$	$\sigma_\phi = 0.25°$
$e_{TX} = 1.26W$	$R_c = 120m$	$\sigma_R = 0.075m$
$e_{RX} = 0.63W$	$R_{HPS} = 55m$	$\sigma_{\boldsymbol{v},x} = \sigma_{\boldsymbol{v},y} = 0.1m$
$e_{DPU} = 1W$	$\mu_{cl} = 0.025$	$\sigma_{\boldsymbol{v},\psi} = 0.1°$
$E_0 = 137592J$	$p_{fa} = 0.01$	$\gamma_{HPS} = 0.95$

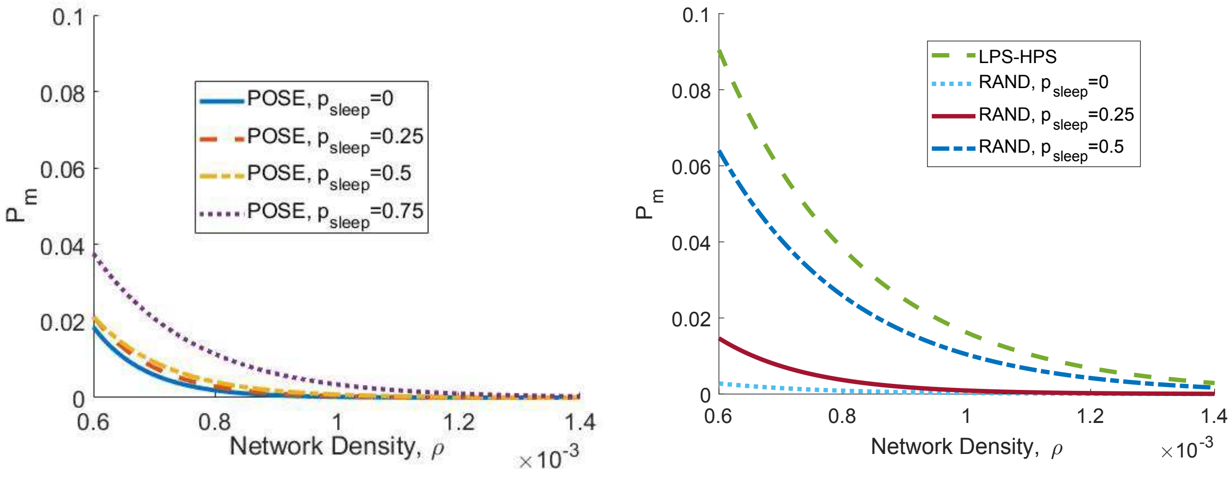


(a) Example of POSE Algorithm PFSA (b) Eample of Random Scheduling (c) Eample of LPS-HPS Scheduling

Figure 3.9: Distributed scheduling methods

First, we obtain the missed detection and percent energy savings characteristics of the POSE network as compared to two distributed scheduling methods, Random Scheduling and LPS-HPS Scheduling. The Random Scheduling method, as shown in Fig. 3.9b, is a distributed method where the sensor nodes probabilistically cycle between actively sensing (*HPS*) and sleeping. Thus, during each time step the nodes sleep with a probability p_{rand}, while they sense the environment in the *HPS* state with a probability $1 - p_{rand}$. When $p_{rand} = 0$, the sensor nodes are always sensing, while for $p_{rand} = 1$ the nodes are always sleeping. The LPS-HPS Scheduling method, as shown in Fig. 3.9c, is another distributed approach that falls under the category of trigger-based activation. Here, the sensor nodes remain in the passive (*LPS*) state until a target is detected. Once a node detects a target, it switches to the active (*HPS*) state for further interrogation. Note that in both the LPS-HPS and Random Scheduling methods, the sensor nodes do not collaborate their target information together.

To present the missed detection characteristics of the POSE, Random, and LPS-HPS scheduling methods, a single target was simulated to travel through the ROI. Each network was run for a varying network density, ρ, between $\rho = [0.6e^{-3}, 1.4e^{-3}]\frac{Nodes}{m^2}$ or $n = [150, 350]$ nodes. The POSE algorithm was simulated for various p_{sleep} values between $p_{sleep} = [0, 0.75]$, while the Random Scheduling method was simulated for a $p_{rand} = [0, 0.5]$. Figure 3.10a presents the missed detection probability, P_m, of the POSE network. This result shows that as the p_{sleep} parameter increases, the probability of missed detection, P_m, increases. This is because the p_{sleep} parameters governs if the node will stay in the *Sleep* state. Therefore, the higher the value of p_{sleep}, the more likely a node within the detection region of the target will be in the *Sleep* state. However, even for



(a) POSE Network

(b) Random and LPS-HPS Scheduling Methods

Figure 3.10: Missed Detection Characteristics

high values of p_{sleep}, the POSE algorithm is able to achieve $< 5\%$ missed detections for low densities. Furthermore, as the network density increases, the probability of missed detection decreases to almost 0. This is because the sensor nodes are able to use information from their neighbors to predict where the target is heading. Then, the nodes are able to pro-actively transition to the *HPS* state to prepare for the target in advance. Therefore, as long as the network density is sufficiently large, there will always be a node aware of the target's location and ready to cover it.

The missed detection characteristics of the Random and LPS-HPS Scheduling methods are presented in Figure 3.10b. As seen, the always sensing network, i.e. RAND, $p_{rand} = 0$, achieves a very low probability of missed detection. Then, as the probability p_{rand} increases, the network starts to miss the target more frequently. This is because in the Random Scheduling network, the sensors are not worried about tracking the target. They are more concerned with covering the entire ROI to facilitate target coverage. Therefore, as the probability p_{rand} increases, this method of scheduling requires a larger network density to achieve the same missed detection characteristics. Additionally, Fig. 3.10b shows the probability of missed detection for the LPS-HPS network. This method produces the highest missed detection probability because the nodes are mainly in the LPS state. These passive sensors have a smaller sensing range than the HPS devices and cannot cover as much of the ROI. Therefore, in order for the LPS-HPS method to achieve as low of a missed detection probability as the other methods, the network must deploy a larger network density.

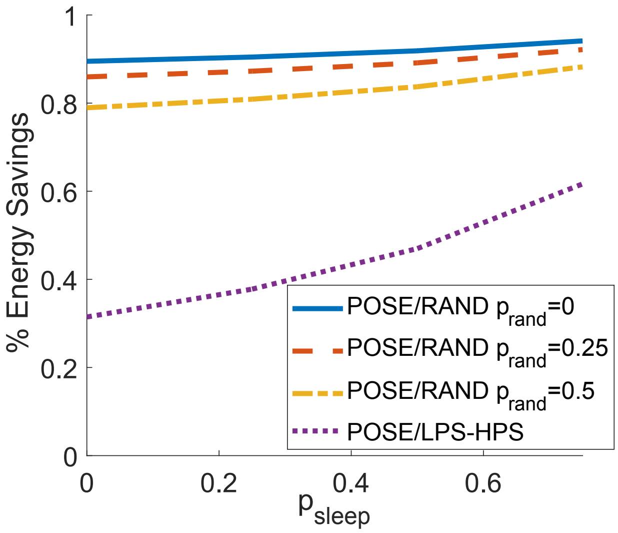
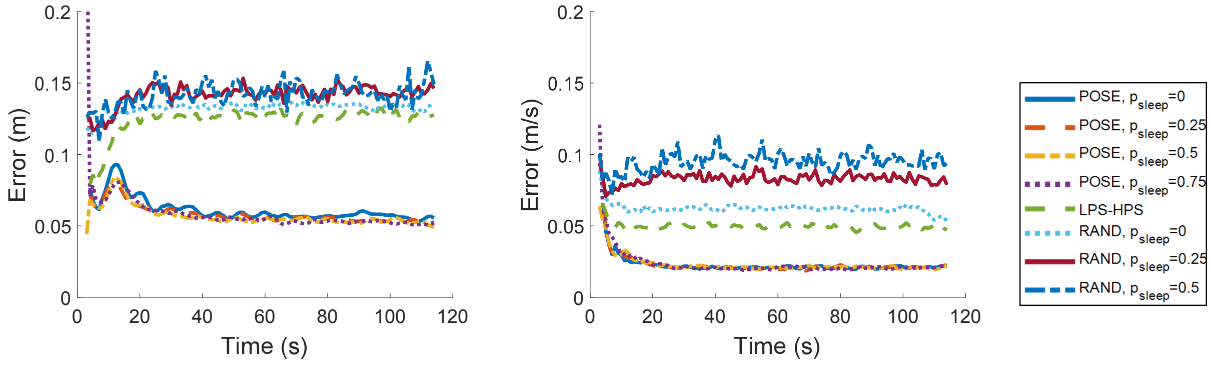


Figure 3.11: Percent Energy Savings of the POSE network

Additionally, as seen in Figures 3.10a and 3.10b, the characteristic curves for most of the p_{sleep} values (say $0 \leq p_{sleep} \leq 0.5$) of the POSE network are similar to RAND, $p_{rand} = 0.25$; thus are close to the always monitoring network. However, the don't go below the always monitoring network which is obvious. The performance of the POSE network approaches the all on scheme as the node density increases. As also seen, the POSE network beats the performance of the LPS-HPS and Random networks with $p_{rand} \geq 0.25$.

While achieving a performance close to that of the always monitoring network, the POSE network results in significant energy savings, as shown in Figure 3.11. This is because only a small fraction of HPS nodes are enabled during each time step around the target. This leads to $> 80\%$ energy savings as compared to the Random Scheduling method with $p_{rand} \leq 0.5$. As compared to the LPS-HPS method, the POSE network is able to achieve $> 30\%$ energy savings. This is because the sensor nodes away from the target are able to conserve energy by cycling between the *Sleep* and *LPS* state, while the LPS-HPS method has all of the sensors away from the target in the *LPS* state. The percent energy savings over the POSE network compared to the other networks is defined as $\% Energy Savings = 1 - \frac{E_{net}(k,POSE)}{E_{net}(k,*)}$, where $E_{net}(k,*)$ is the total network energy consumption for the network $* \in \{$POSE, RAND $p_{rand} = 0$, RAND $p_{rand} = 0.25$, RAND $p_{rand} = 0.5$, LPS-HPS$\}$. Thus, the following trade off exists: for energy savings, the value of p_{sleep} should be large, while for reducing the P_m, the value of p_{sleep} should be small. Based on the results in Figure 3.10a,



(a) Position RMSE

(b) Velocity RMSE

Figure 3.12: Root Mean Squared Error

the network density of $\rho = 1.4e^{-3}$ is chosen for further validation studies.

Next, the tracking performance of the three algorithm was studied and is presented in Figure 3.12. As seen in Fig. 3.12, the parameter p_{sleep} does not affect the Root Mean Squared Error (RMSE) in both position and velocity. This is because even though some sensors might be in the *Sleep* state around the target, a cluster of *HPS* state nodes is still tracking it. Additionally, it can be seen that the RMSE of the POSE algorithm is much lower then the other distributed methods. This is because the *LPS* and *HPS* nodes in the POSE network are able to collaborate and improve their state estimates through distributed fusion. Therefore, the POSE algorithm is able to achieve accurate state estimation, low missed detection rates, and significant energy savings.

3.5 Conclusion and Limitations

This chapter presented a novel distributed supervisory control algorithm, called *Prediction-based Opportunistic Sensing* (POSE), that facilitates probabilistic control of multi-modal sensor nodes. Each sensor node is controlled using a *Probabilistic Finite State Automaton* (PFSA), which dynamically adapts its state transition probabilities to control the communication and sensing devices on the node based on the targets information. Simulation results verify that the POSE network performs opportunistic sensing, significantly reduces energy consumption, achieves low missed detection rates, and accurate state estimate as compared to Random Scheduling and LPS-HPS methods.

However, there are two main drawbacks of the POSE algorithm that can improve the energy-efficiency of the network. First, the *HPS* sensors around that target consist of every sensor node located within the detection region of the target. Although this is providing accurate state estimation, the energy consumption around the target is far too high due to redundant *HPS* state nodes. Therefore, if a distributed sensor selection method is implemented to select a subset of the sensors around the target, energy wastage would be minimized. The second major drawback of this approach is that the network assumes that all targets located within the ROI are of interest to the network operators. In practical applications, many different types targets will pass through the network and only a hand full of them will be of interest to the network operator. Therefore, if the target class was used to control the number of sensors active in the *HPS* state around the target, significant energy could be saved.

To overcome the above limitations, the *Prediction-based Opportunistic Sensing using Distributed Classification, Clustering and Control* (POSE.3C) algorithm was developed and is discussed in the next chapter.

CHAPTER 4

PREDICTION-BASED OPPORTUNISTIC SENSING USING DISTRIBUTED CLASSIFICATION, CLUSTERING AND CONTROL (POSE.3C)

4.1 3C Network Autonomy Introduction

Chapter 3 developed an advanced *distributed supervisory control* approach that probabilistically controls the multi-modal devices on each node. This approach ensures that the nodes around the target autonomously activate their HPS (e.g. active) devices to track the target, while the nodes away from the target either enable their LPS (e.g. passive) devices to stay aware, or *Sleep* to preserve energy.

This supervisory control approach requires predictive intelligence of a target's whereabouts to form clusters of *HPS* state nodes around the target. In this regard, the first issue addressed in this chapter is *distributed clustering* via efficient sensor node selection. Current clustering approaches typically select sensors based on their distance to the target or detection capability [31]. However, in applications where targets may frequently travel in the same section of the network multiple times (e.g., multiple targets trespassing through a lane), these approaches will select the same sensor nodes again and again, thus depleting their energies and creating holes or energy depleted lanes in the network. Thus, uniform spatial distribution of remaining energy resources around regions with frequent target visits is a desired feature. In our approach, we pursue a multi-step filtering process for node selection which considers factors including distance to the target, the remaining energies, and the geometric distribution around the target. This clustering process enables the network to track the target accurately while maintaining a uniform energy distribution, hence minimizing holes and further improving longevity.

Furthermore, it is observed that within the above opportunistic sensing paradigm, energy wastage may still occur if clusters are formed around a *Target Not of Interest* (TNOI) to the mission operator. Therefore, it is desired that *distributed classification* is incorporated into the control strategy, such that energy resources are dynamically allocated primarily around a *Target of Interest* (TOI). For example, in a border surveillance application, the TOIs could be humans and vehicles, while the TNOIs could be animals. In our approach, we adapt the cluster size based on the classification decisions, such that $N_{sel} > 1$ sensor nodes are activated around a TOI to improve the estimation accuracy via fusion, while only 1 sensor node is activated around a TNOI to maintain awareness. This classification-enabled network control approach drastically improves the network lifetime by performing opportunistic sensing only around mission specific events.

To address the above issues of *Classification→Clustering→Control* (3C network autonomy), this chapter presents a distributed algorithm, called *Prediction-based Opportunistic Sensing using Distributed Classification, Clustering and Control* (POSE.3C), which manages the different multimodal devices on each sensor node for opportunistic energy-efficient operation of the network. As shown in Fig. 4.1, a distributed supervisor designed as a PFSA is embedded on each node to enable/disable its different sensing and communication devices. The states of the PFSA include: 1) *Sleep*, 2) *LPS*, and 3) *HPS*. The *Sleep* state disables all devices to minimize energy consumption. The *LPS* state enables the LPS devices for target detection, while the *HPS* state enables the HPS devices for state estimation and classification of the target. The TX and RX are turned on in both *LPS* and *HPS* states for sensor collaboration.

The state transition probabilities of the PFSA are dynamically updated based on the information regarding the target's class and predicted location. This information could be either measured and computed locally by the node and/or received from the neighbors. If it is received, the nodes will perform distributed fusion to identify the target's class (TOI or a TNOI) and predict its state during

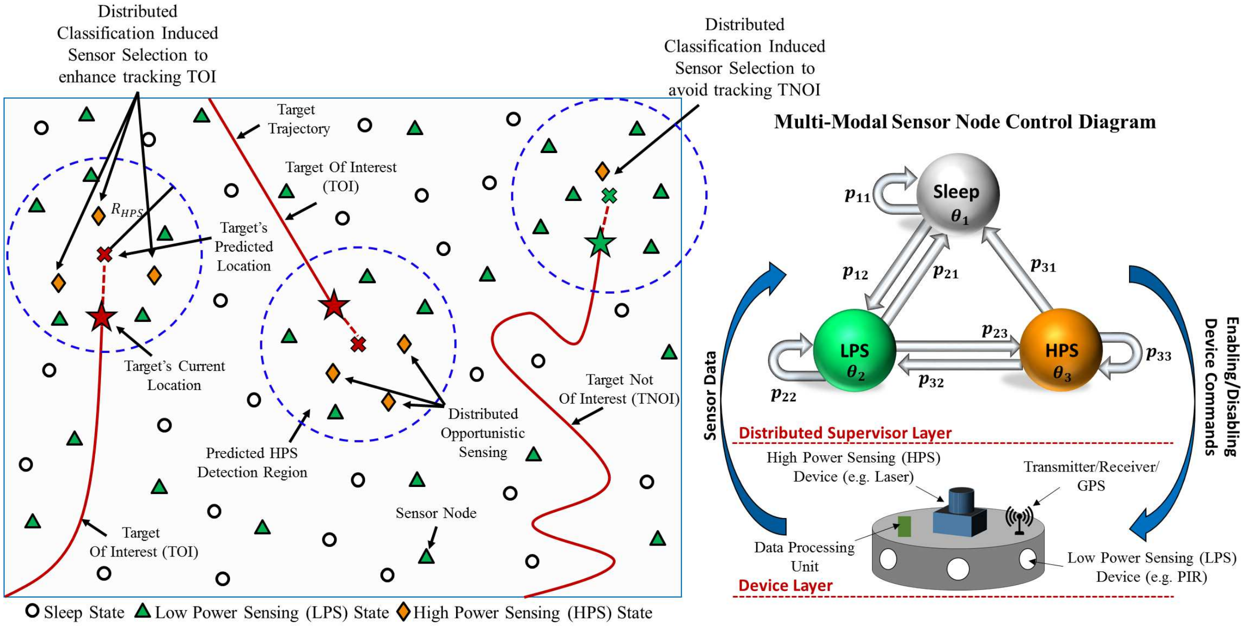

Distributed Classification Induced Sensor Selection to enhance tracking TOI
Distributed Classification Induced Sensor Selection to avoid tracking TNOI
Target Trajectory
Target Of Interest (TOI)
R_{HPS}
Target's Predicted Location
Target's Current Location
Predicted HPS Detection Region
Distributed Opportunistic Sensing
Target Not Of Interest (TNOI)
Sensor Node
Target Of Interest (TOI)
○ Sleep State ▲ Low Power Sensing (LPS) State ◆ High Power Sensing (HPS) State
Multi-Modal Sensor Node Control Diagram
Sleep θ_1
LPS θ_2
HPS θ_3
p_{11}
p_{12}
p_{21}
p_{31}
p_{22}
p_{23}
p_{32}
p_{33}
Sensor Data
Enabling/Disabling Device Commands
Distributed Supervisor Layer
High Power Sensing (HPS) Device (e.g. Laser)
Transmitter/Receiver/ GPS
Data Processing Unit
Low Power Sensing (LPS) Device (e.g. PIR)
Device Layer

Figure 4.1: Illustration of the POSE.3C algorithm with a PFSA-based distributed supervisor acting on each node.

the next time interval. This fused information is then used to form clusters of optimal sensors. The selected nodes probabilistically transition to the *HPS* state while nodes not selected cycle between the low power consuming states (i.e *Sleep* and *LPS*) to stay aware while preserving energy. Fig. 4.1 illustrates this approach with three targets traveling through the network. As illustrated, $N_{sel} = 3$ sensors are activated in the *HPS* state around the TOIs, while only 1 sensor is activated around the TNOI. Ideally, 0 sensors should be activated around the TNOI, however, if there is not a *HPS*state node tracking the TNOI, then all of the *LPS* state nodes surrounding the target will not receive any information about it. Therefore, following the *LPS* state control design in Chapter 3.3.2, the *LPS* state nodes will immediately switch to the *HPS* state and redundant sensor nodes will be active in a similar manner as the POSE algorithm. Therefore, if 1 *HPS* state node is actively tracking TNOI, then the *LPS* state nodes will be alerted that a TNOI is present and will not waste their energy by transition to the *HPS* state. Thus, a POSE.3C network opportunistically self-adapts around the TOIs, providing significant energy savings.

The main contributions of this Chapter are as follows:

- Development of a distributed algorithm that facilitates 3C network autonomy with the following attributes:

(i) distributed classification and utilization of the target class information to govern the cluster size for tracking,

(ii) distributed clustering by fusing the target state estimates, predicting the next state, and performing sensor node selection around the target's predicted state based on the targets class, and

(iii) distributed supervisory control to enable/disable multi-modal sensing devices for energy-efficiency.

- Theoretical analysis of the network's performance characteristics: (i) Expected energy consumption, (ii) Expected network lifetime, (iii) Probability of missed detection for a target birth and for a mature target.

4.2 Related Work

The following subsections describe the existing classification and scheduling methods employed in sensor networks.

4.2.1 Classification Methods in Networks

Several pattern recognition methods, such as κ-Nearest Neighbors, Support Vector Machines, and Symbolic Feature Extraction, are popular for computing classification decisions based on sensor data. Examples of these approaches include classifying vehicles, soldiers, and pedestrians [73], trespassers [7, 74], mines [75], submarines, and sea animals [24]. Typically, these approaches require the nodes to be equipped with sensors that measure acoustic signals, magnetic fields, and/or images, that provide distinguishing features between target classes. Several target tracking networks only measure the range and azimuth of the target, thus kinematic feature based classification methods were developed. These methods compute the posteriori probability of the target motion model given the observed track, known as Joint Tracking and Classification [32, 76]. These methods are typically used in radar/sonar applications (e.g., classifying commercial aircraft vs. fighter jets) and have a high computational complexity which may not be suitable for DSN. Since centralized classification is impractical for large DSN, most networks perform classification at the cluster head by fusing local decisions or features that were computed on each sensor node [24, 73, 74].

The above approaches only deal with the classification problem and do not address the integrated 3C network autonomy problem which is the focus of this chapter.

4.2.2 Network Control and Scheduling

Typical methods of network control deal with minimizing the number of active nodes around the target via sensor selection. This is achieved by Centralized or CH-based approaches. CH-based approaches are more popular due to their feasibility in large networks, where the Cluster Head (CH) selects the optimal set of sensors to track the target [23, 32, 50, 64]. The optimization usually consists of maximizing/minimizing one of the following cost functions: probability of detection [27], tracking accuracy [77], energy [30], Kullback-Liebler distance [31], or other information theoretic measures [32, 78]. The above methods are not practical for DSN due to the heavy communication and computation requirements of searching the entire network/cluster. Furthermore, if a CH were to fail, the entire cluster would stop receiving scheduling decisions, rendering a large gap in the ROI. This requires implementation of dynamic CH election schemes [23]. Thus, for DSN, Kaplan [29] proposed a local node selection algorithm based on a cost function that minimizes the Mean Squared Error of the target state. This approach was extended by utilizing the innovation of the estimate [79] and using mutual information [80].

Although these methods performed sensor selection, they neither performed the multi-modal probabilistic control of the selected nodes nor addressed the integrated 3C network autonomy problem for energy-efficient target tracking.

4.3 POSE.3C Algorithms

As shown in Fig. 4.1, each sensor node is controlled by a distributed PFSA-based supervisor as defined in Chapter 3.3.

Consider a sensor node $s_i \in \mathcal{S}$. The operations within each state of its distributed PFSA-based

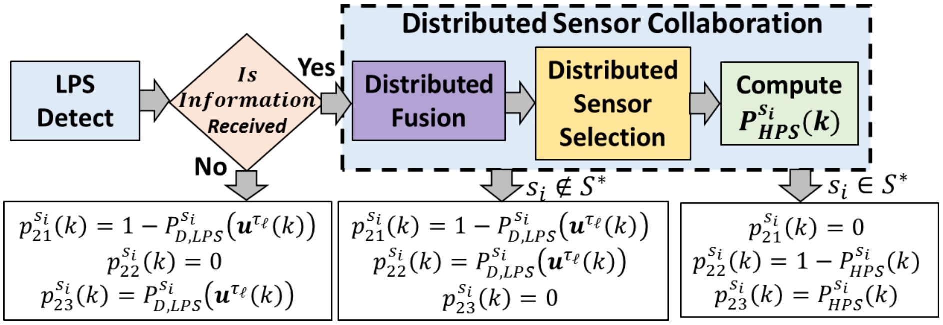

LPS Detect
Is Information Received
Yes
No
Distributed Sensor Collaboration
Distributed Fusion
Distributed Sensor Selection
Compute $P^{s_i}_{HPS}(k)$
$s_i \notin S^*$
$s_i \in S^*$
$p^{s_i}_{21}(k) = 1 - P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$
$p^{s_i}_{22}(k) = 0$
$p^{s_i}_{23}(k) = P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$
$p^{s_i}_{21}(k) = 1 - P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$
$p^{s_i}_{22}(k) = P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$
$p^{s_i}_{23}(k) = 0$
$p^{s_i}_{21}(k) = 0$
$p^{s_i}_{22}(k) = 1 - P^{s_i}_{HPS}(k)$
$p^{s_i}_{23}(k) = P^{s_i}_{HPS}(k)$

Figure 4.2: Low Power Sensing State Algorithm

supervisor and how the node updates its state transition probabilities are discussed below.

4.3.1 Sleep State Description

The *Sleep* state, θ_1, is designed to minimize the energy consumption of the node s_i when the target is away or if the node is not selected to track the target. It disables all sensing and communication devices on the node except the clock. After every time interval ΔT, the node can make a transition to the *LPS* state with a probability $p^{s_i}_{12}(k) = 1 - p_{sleep}$ or it can stay in the *Sleep* state with a probability $p^{s_i}_{11}(k) = p_{sleep}$, where p_{sleep} is a design parameter.

4.3.2 Low Power Sensing (LPS) State Description

The *LPS* state, θ_2, is designed to conserve energy while enabling target detection. In this state, the LPS devices, the DPU, TX, and RX are powered on. Fig. 4.2 shows the flowchart for the algorithm within the *LPS* state.

• ***Target Detection***: Here a target, τ_ℓ, could be detected either by: (i) using the LPS devices or by (ii) fusing the information received from neighbors. If a target is present, then the sensor node s_i detects it with a probability $P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$ as per Eq. (2.5).

• ***Distributed Collaboration***: Next, it checks if it has received any information from its neighbors. Let $\mathcal{N}^{s_i}_{HPS} \subseteq \mathcal{N}^{s_i}$ be the set of *HPS* state sensors that have broadcasted the target state and class information to s_i. If information is received, i.e. $\mathcal{N}^{s_i}_{HPS} \neq \emptyset$, then the node s_i uses a collaboration algorithm (see Section 4.4 for details), which fuses the received information to obtain the fused state prediction and classification decision. The fused information is used to form a cluster

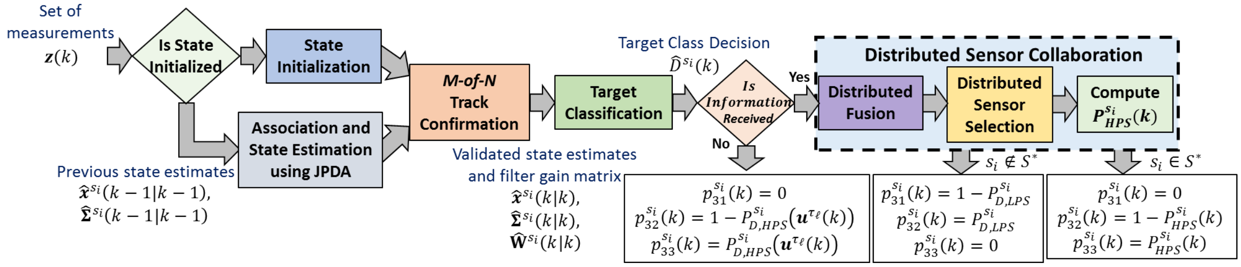

Set of measurements
$\mathbf{z}(k)$
Is State Initialized
State Initialization
Association and State Estimation using JPDA
Previous state estimates
$\hat{\mathbf{x}}^{s_i}(k-1|k-1)$,
$\hat{\mathbf{\Sigma}}^{s_i}(k-1|k-1)$
M-of-N Track Confirmation
Validated state estimates and filter gain matrix
$\hat{\mathbf{x}}^{s_i}(k|k)$,
$\hat{\mathbf{\Sigma}}^{s_i}(k|k)$,
$\hat{\mathbf{W}}^{s_i}(k|k)$
Target Classification
Target Class Decision
$\hat{D}^{s_i}(k)$
Is Information Received
Yes
No
Distributed Sensor Collaboration
Distributed Fusion
Distributed Sensor Selection
Compute $P^{s_i}_{HPS}(k)$
$s_i \notin S^*$
$s_i \in S^*$
$p^{s_i}_{31}(k) = 0$
$p^{s_i}_{32}(k) = 1 - P^{s_i}_{D,HPS}(\mathbf{u}^{\tau_\ell}(k))$
$p^{s_i}_{33}(k) = P^{s_i}_{D,HPS}(\mathbf{u}^{\tau_\ell}(k))$
$p^{s_i}_{31}(k) = 1 - P^{s_i}_{D,LPS}$
$p^{s_i}_{32}(k) = P^{s_i}_{D,LPS}$
$p^{s_i}_{33}(k) = 0$
$p^{s_i}_{31}(k) = 0$
$p^{s_i}_{32}(k) = 1 - P^{s_i}_{HPS}(k)$
$p^{s_i}_{33}(k) = P^{s_i}_{HPS}(k)$

Figure 4.3: High Power Sensing State Algorithm

of optimal sensors, $\mathcal{S}^*$, to track the target. If $s_i \in \mathcal{S}^*$, then it computes a probability $P^{s_i}_{HPS}(k)$ (see Eq. (3.12)) to transition to the *HPS* state. On the other hand, if no information is received, i.e. $\mathcal{N}^{s_i}_{HPS} = \emptyset$, then the node relies on its own detection probability $P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$ to transition to the *HPS* state.

• ***Updating the State Transition Probabilities***: The node s_i updates it's PFSA probabilities as follows:

- *if* ($\mathcal{N}^{s_i}_{HPS} \neq \emptyset$ and $s_i \in \mathcal{S}^*$), *node s_i is selected, then*:
  $p^{s_i}_{21}(k) = 0$; $p^{s_i}_{22}(k) = 1 - P^{s_i}_{HPS}(k)$; $p^{s_i}_{23}(k) = P^{s_i}_{HPS}(k)$

- *if* ($\mathcal{N}^{s_i}_{HPS} \neq \emptyset$ and $s_i \notin \mathcal{S}^*$), *node s_i is not selected but has received information, then*:
  $p^{s_i}_{21}(k) = 1 - P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$; $p^{s_i}_{22}(k) = P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$; $p^{s_i}_{23}(k) = 0$

- *if* ($\mathcal{N}^{s_i}_{HPS} = \emptyset$), *node s_i does not receive information, then*:
  $p^{s_i}_{21}(k) = 1 - P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$; $p^{s_i}_{22}(k) = 0$; $p^{s_i}_{23}(k) = P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$

4.3.3 High Power Sensing (HPS) State Description

The *HPS* state, θ_3, is designed to estimate the target's state and class by using measurements from its HPS devices and broadcast this information to alert its neighbors of a targets presence. Here, the HPS devices, DPU, TX and RX are all enabled. Fig. 4.3 shows the flowchart for the algorithm within the *HPS* state.

• ***Data Association and State Estimation***: In *HPS* state, the node s_i first receives a set of measurements, $\mathbf{z}(k)$, from its HPS devices. Subsequently, the previous state estimate $\hat{\mathbf{x}}^{s_i}(k-1|k-1)$, $\hat{\mathbf{\Sigma}}^{s_i}(k-1|k-1)$ are updated using the *Joint Probabilistic Data Association* (JPDA) method [48] to generate $\hat{\mathbf{x}}^{s_i}(k|k)$, $\hat{\boldsymbol{\Sigma}}^{s_i}(k|k)$ (please see Appendix B for details). If the received measurements do not associate to the previous state estimate (e.g., when the target state is not initialized), then the node s_i must first perform state initialization [71].

The measurements received may contain false alarms, due to clutter, which can generate false tracks at each node. To ensure that false tracks do not propagate throughout the network, node s_i utilizes the *M-of-N Track Confirmation Logic* [72] to allow the network to be robust to false alarms. This approach ensures that M out of N consecutive time steps have measurements that are associated to a target state estimate before the node confirms that it is not a false track. Furthermore, once the target track has been confirmed, the node can drop the track if M consecutive time steps have measurements that do not associate to it.

• ***Target Classification***: Next, node s_i performs target classification to determine the target class. To keep our control algorithm general, it is assumed that the network designer has developed a classifier (similar to those reported in Section 4.2.1) for the particular application and its performance is represented by a *Confusion Matrix* B, as shown in Table 4.1. Then, node s_i will classify the target as TOI with a probability

Table 4.1: Confusion Matrix

		Estimated Class	
		TOI	TNOI
True Class	TOI	b_{11}	b_{12}
	TNOI	b_{21}	b_{22}

$$\hat{P}^{s_i}_{TOI}(k) = \begin{cases} \frac{b_{11}}{b_{11}+b_{12}} & \text{Given TOI} \\ \frac{b_{21}}{b_{21}+b_{22}} & \text{Given TNOI} \end{cases} \tag{4.1}$$

which leads to a class decision $\hat{D}^{s_i}(k) \in \{0, 1\}$, where 0 and 1 correspond to TNOI and TOI, respectively.

• ***Distributed Collaboration***: Next, if a target has been detected, then node s_i broadcasts its target state estimates $\hat{\mathbf{x}}^{s_i}(k|k)$ and $\hat{\boldsymbol{\Sigma}}^{s_i}(k|k)$, the filter gain matrix $\hat{\mathbf{W}}^{s_i}(k)$, and the classification decision $\hat{D}^{s_i}(k)$. Since s_i is in the *HPS* state and has broadcasted information to it's neighbors, we define $\mathcal{N}^{s_i}_{HPS} = \mathcal{N}^{s_i}_{HPS} \cup \{s_i\}$. However, if s_i has not confirmed a target track, then $\mathcal{N}^{s_i}_{HPS}$ does

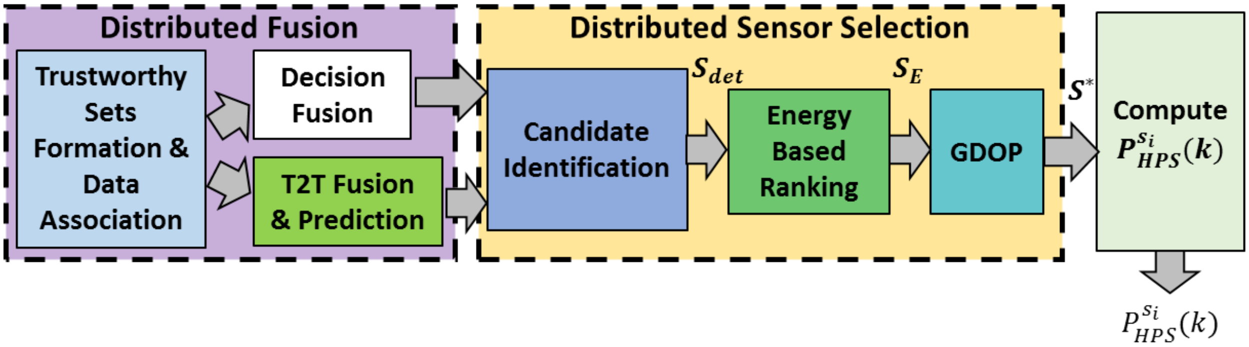


Figure 4.4: Distributed Sensor Collaboration Algorithm

not include s_i. Then, if $\mathcal{N}^{s_i}_{HPS} \neq \emptyset$, it will run the collaboration algorithm (please see Section 4.4 for details) in the same manner as in the *LPS* state. This generates the fused state and class decision, which are then used to form the cluster of optimal sensors $\mathcal{S}^*$ to track the target during the next time step. If $s_i \in \mathcal{S}^*$, then it computes $P^{s_i}_{HPS}(k)$ in Eq. (3.12) to stay in the *HPS* state.

• ***Updating the State Transition Probabilities***: Finally, the PFSA probabilities are updated as follows:

- *if* ($\mathcal{N}^{s_i}_{HPS} \neq \emptyset$ and $s_i \in \mathcal{S}^*$), *node* s_i *is selected, then*:

  $p^{s_i}_{31}(k) = 0$; $p^{s_i}_{32}(k) = 1 - P^{s_i}_{HPS}(k)$; $p^{s_i}_{33}(k) = P^{s_i}_{HPS}(k)$

- *if* ($\mathcal{N}^{s_i}_{HPS} \neq \emptyset$ and $s_i \notin \mathcal{S}^*$), *node* s_i *is not selected but has received information or verified a target track, then*:

  $p^{s_i}_{31}(k) = 1 - P^{s_i}_{D,HPS}(\mathbf{u}^{\tau_\ell}(k)$; $p^{s_i}_{32}(k) = P^{s_i}_{D,HPS}(\mathbf{u}^{\tau_\ell}(k)$; $p^{s_i}_{33}(k) = 0$

- *if* ($\mathcal{N}^{s_i}_{HPS} = \emptyset$), *node* s_i *has not verified a track or received information, then*:

  $p^{s_i}_{31}(k) = 0$; $p^{s_i}_{32}(k) = 1 - P^{s_i}_{D,HPS}(\mathbf{u}^{\tau_\ell}(k))$; $p^{s_i}_{33}(k) = P^{s_i}_{D,HPS}(\mathbf{u}^{\tau_\ell}(k)$

4.4 Distributed Sensor Collaboration

The distributed sensor collaboration algorithm, shown in Fig. 4.4, consists of distributed fusion, sensor selection, and computation of the *HPS* state transition probability, which are described below:

4.4.1 Distributed Fusion

Similar to Chapter 3, distributed fusion consists of combining the received state estimates together into one estimate per target located in the nodes communication neighborhood. However, in this chapter, classification decision are also transmitted. Therefore, the distributed fusion algorithm must incorporate a method to combine target class decisions as well.

Consider a node s_i which could be in the *LPS* or *HPS* state. The information ensemble it receives consists of

$$\hat{\mathbf{I}}^{s_i}(k) = \big\{\big(\hat{\mathbf{x}}^{s_j}(k|k), \hat{\mathbf{\Sigma}}^{s_j}(k|k), \hat{\mathbf{W}}^{s_j}(k), \hat{D}^{s_j}(k)\big), \forall s_j \in \mathcal{N}_{HPS}^{s_i}\big\}, \tag{4.2}$$

where $\hat{\mathbf{x}}^{s_j}(k|k)$, $\hat{\mathbf{\Sigma}}^{s_j}(k|k)$, $\hat{\mathbf{W}}^{s_j}(k)$, and $\hat{D}^{s_j}(k)$ correspond to the state estimate, covariance, filter gain, and class decision, of the node s_j at time k.

However, due to noise and other factors, the information received must first be validated to ensure that it is accurate and reliable before processing. This is done by forming a set of trustworthy neighbors $\mathcal{N}_T^{s_i} \subseteq \mathcal{N}_{HPS}^{s_i}$ by evaluating the sum of the position error as follows

$$\mathcal{N}_T^{s_i} = \{s_j \in \mathcal{N}_{HPS}^{s_i} : Trace(\mathbf{H}(k)\hat{\mathbf{\Sigma}}^{s_j}(k|k)\mathbf{H}(k)') \leq \xi\} \tag{4.3}$$

where $\mathbf{H}(k)$ is the Jacobian of the measurement model defined in Chapter 2.2 and ξ is the maximum tolerance of the estimate. In this work, $\xi = \frac{R_{HPS}^2\sigma_\phi^2 + \sigma_R^2}{2}$, where σ_ϕ and σ_R are the standard deviations in the azimuth and range measurements of the HPS sensor. For more details, see Chapter 3.3.2. Finally, node s_i receives the following trustworthy information:

$$\hat{\mathbf{I}}_T^{s_i}(k) = \big\{\big(\hat{\mathbf{x}}^{s_j}(k|k), \hat{\mathbf{\Sigma}}^{s_j}(k|k), \hat{\mathbf{W}}^{s_j}(k), \hat{D}^{s_j}(k)\big), \forall s_j \in \mathcal{N}_T^{s_i}\big\}, \tag{4.4}$$

Next, the trustworthy information is associated to ensure that it is related to the same target. In this work, the *Track-to-Track Association Method* (T2TA) [69] is used for this purpose. Details of the T2TA method are presented in Chapter 3.3.2. This method associates the trustworthy information into C different groups which correspond to the C different targets that could be present within the node s_i's neighborhood; thus forming the information ensembles: $\hat{\mathbf{I}}_T^{s_i,c}(k) \subseteq \hat{\mathbf{I}}_T^{s_i}(k)$, where $c = 1, ..., C$. Subsequently, for each c, the state information in $\hat{\mathbf{I}}_T^{s_i,c}(k)$ is fused to form a single state ($\hat{\mathbf{x}}^{s_i,c}(k|k)$) and covariance ($\hat{\mathbf{\Sigma}}^{s_i,c}(k|k)$) estimate, using the *Track-to-Track Fusion* (T2TF) algorithm [70]. Details of the T2TF method are presented in Chapter 3.3.2. Once all the received state information are fused, node s_i computes a one-step prediction [45] of each target's state using the Extended Kalman Filter:

$$
\begin{aligned}
\hat{\mathbf{x}}^{s_i,c}(k+1|k) &= \mathbf{f}(\hat{\mathbf{x}}^{s_i,c}(k|k), k) \\
\hat{\mathbf{\Sigma}}^{s_i,c}(k+1|k) &= \mathbf{F}(k)\hat{\mathbf{\Sigma}}^{s_i,c}(k|k)\mathbf{F}(k)' + \mathbf{Q},
\end{aligned}
\tag{4.5}
$$

where $\mathbf{F}(k)$ is the Jacobian of the state transition matrix evaluated at $\hat{\mathbf{x}}^{s_i,c}(k|k)$ and $\mathbf{Q}$ is the process noise covariance matrix. This predicted state is used in the distributed sensor selection algorithm to select the sensors to track the target during next time step for opportunistic sensing.

Furthermore, the associated target class decisions in $\hat{\mathbf{I}}_T^{s_i,c}(k)$ are also fused together as follows.

$$
\hat{D}^{s_i,c}(k) = \begin{cases} 1 & if \frac{1}{|\hat{\mathbf{I}}_T^{s_i,c}(k)|} \sum_{\hat{D}^{s_j}(k) \in \hat{\mathbf{I}}_T^{s_i,c}(k)} \hat{D}^{s_j}(k) \geq 0.5 \\ 0 & else \end{cases}
\tag{4.6}
$$

which will be used to determine the number of sensors selected to track the target depending on its class.

4.4.2 Distributed Sensor Selection

The Distributed Sensor Selection algorithm employs a multi-step filtering process performed at each sensor node to identify the optimal sensors to track the target in the *HPS* state during the next time step. These steps are described below.

Step 1: The node s_i first locates the sensors within it's neighborhood that can detect a target c in the detection region around the target's predicted position during the next time step to form a candidate set $\mathcal{S}_{det}$, such that

$$
\mathcal{S}_{det} = \{s_j \in (\mathcal{N}^{s_i} \cup s_i) : ||\mathbf{u}^{s_j} - \hat{\mathbf{z}}^{s_i,c}(k+1|k)|| \leq R_{HPS}\}
\tag{4.7}
$$

where $\hat{\mathbf{z}}^{s_i,c}(k+1|k) = \mathbf{h}(\hat{\mathbf{x}}^{s_i,c}(k+1|k), k)$.

Step 2: Then, $N'_{sel} \geq N_{sel}$ sensors are filtered from $\mathcal{S}_{det}$ that have the highest energy remaining. At this state, if node $s_i \in \mathcal{S}_{det}$, then it will broadcast it's energy consumption $E^{s_i}(k)$ to it's neighbors. Then, the node s_i computes the predicted remaining energy for each node in $\mathcal{S}_{det}$ as follows:

$$E_R^{s_j} = 1 - \left(\frac{E^{s_j}(k) + E_{HPS}\Delta T}{E_0} \right), \ \forall s_j \in \mathcal{S}_{det} \tag{4.8}$$

where $E^{s_j}(k)$ is the energy already consumed by s_j; $E_{HPS} = (e_{HPS} + e_{RX} + e_{TX} + e_{DPU})$ is the energy consumed per unit time in the ***HPS*** state; ΔT is the time duration for which the node s_j will be in the ***HPS*** state if selected; and E_0 is the node's initial energy. Next, each sensor $s_j \in \mathcal{S}_{det}$ is ranked in descending order based on their energy remaining. Then the set $\mathcal{S}_E \subseteq \mathcal{S}_{det}$ is selected to consist of the top ranked N'_{sel} sensors. Note that N'_{sel} is determined based on the target class decision as follows:

$$N'_{sel} = \begin{cases} \geq N_{sel} & if \hat{D}^{s_i}(k) = 1 \\ 1 & if \hat{D}^{s_i}(k) = 0 \end{cases} \tag{4.9}$$

Therefore, if the target is a TNOI, the node with the highest energy remaining is selected to track the TNOI. Whereas for a TOI, a designed value $\geq N_{sel}$ is selected to reduce the computational complexity. If $N'_{sel} = |\mathcal{S}_{det}|$, then the sensor selection process will select sensors purely based on their geometric diversity and to minimize the state covariance error. However, if $N'_{sel} = N_{sel}$, the sensors selected will be purely based on energy remaining. Therefore, by choosing a value between $N_{sel} < N'_{sel} < |\mathcal{S}_{det}|$, the sensor selection process will select sensors are geometrically diverse and allow for uniform energy depletion.

Step 3: Finally, N_{sel} sensors are selected from the set $\mathcal{S}_E$ that are geometrically distributed around the target's predicted position using the reciprocal of the *Geometric Dilution Of Precision* (GDOP) measure [29]. This measure selects nodes that minimize the measurement covariance error to ensure accurate tracking accuracy and is defined as

$$\mu(\widetilde{\mathcal{S}}) = \frac{det(\mathbf{J}(\widetilde{\mathcal{S}}))}{trace(\mathbf{J}(\widetilde{\mathcal{S}}))}, \tag{4.10}$$

$$\mathbf{J}(\widetilde{\mathcal{S}}) = \sum_{s_j \in \widetilde{\mathcal{S}}} \frac{1}{\sigma_\phi^2 r_{s_j}^2} \begin{bmatrix} sin^2\phi_{s_j} & -sin\phi_{s_j} cos\phi_{s_j} \\ -sin\phi_{s_j} cos\phi_{s_j} & cos^2\phi_{s_j} \end{bmatrix},$$

where r_{s_j} is the range of sensor s_j to the target's predicted position; ϕ_{s_j} is the azimuth angle between sensor s_j and the target's predicted position; and $\widetilde{\mathcal{S}} \subseteq \mathcal{S}_E$, s.t. $|\widetilde{\mathcal{S}}| = N_{sel}$. The optimal set $\mathcal{S}^* \subseteq \mathcal{S}_E \subseteq \mathcal{S}_{det}$ is generated as

$$\mathcal{S}^* = \underset{\widetilde{\mathcal{S}} \subseteq \mathcal{S}_E}{argmax}(\mu(\widetilde{\mathcal{S}})) \tag{4.11}$$

Note that $\mathcal{S}^*$ is computed for each target track c. Furthermore, note that N_{sel} adapts based on the target class decision. If $\hat{D}^{s_i}(k) = 1$, then $N_{sel} > 1$, and if $\hat{D}^{s_i}(k) = 0$, then $N_{sel} = 1$.

4.4.3 Computation of the *HPS* Transition Probability

If $s_i \in \mathcal{S}^*$ for any target track, then it should transition to the *HPS* state to track the target during the next time step. Node s_i first computes its expected probability of detecting the target based on target's predicted position according to Equations 3.11 and 3.12. This is then used to transition to the *HPS* state as described in the computation of state transition probabilities in Section 4.3.

4.5 Network Characteristics

This section presents the characteristics of the POSE.3C network in terms of the expected energy consumption, network lifetime, and the missed detection probabilities.

Consider that m TOIs (or TNOIs) are present in Ω during a time interval ΔT. We partition Ω

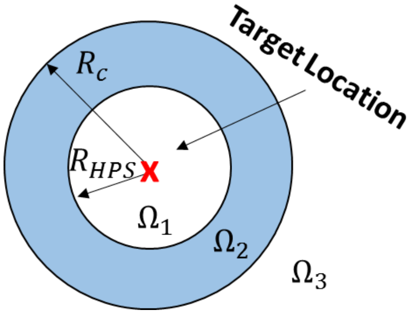


Figure 4.5: Illustration of regions around a single target.

into three regions as follows:

$$\begin{aligned}
\Omega_1 &= \bigcup_{\tau_\ell}\Big\{(x,y) : ||(x,y)-\mathbf{u}^{\tau_\ell}|| \le R_{HPS}\Big\} \\
\Omega_2 &= \bigcup_{\tau_\ell}\Big\{(x,y) : ||(x,y)-\mathbf{u}^{\tau_\ell}|| \le R_c\Big\} \setminus \Omega_1 \\
\Omega_3 &= \Omega \setminus (\Omega_1 \cup \Omega_2)\,; \qquad (4.12)
\end{aligned}$$

where Ω_1 is the detection region around the targets, Ω_2 is the region outside of the detection region but within the communication region around the targets, and Ω_3 is the remaining region away from the targets, as shown in Fig. 4.5. The areas of these regions are denoted by A_{Ω_1}, A_{Ω_2}, and A_{Ω_3}. To be conservative, we assume that the targets have disjoint detection regions. Furthermore, Ω_1 consists of two sets of sensors: Ω_1^* that contains the sensors selected for tracking, and $\Omega_1^{*'}$ that contains the sensors not selected for tracking.

The energy consumed per node in a ΔT time interval in the above regions depends on its state and is given as

$$\begin{aligned}
E_{Sleep} &= e_{clock}\Delta T, \\
E_{LPS}^{\Omega_1} &= (e_{LPS}+e_{TX}+e_{RX}+e_{DPU})\Delta T, \\
E_{HPS}^{\Omega_1} &= (e_{HPS}+e_{TX}+e_{RX}+e_{DPU})\Delta T, \\
E_{LPS}^{\Omega_2,\Omega_3} &= (e_{LPS}+e_{RX}+e_{DPU})\Delta T, \text{ and} \\
E_{HPS}^{\Omega_2,\Omega_3} &= (e_{HPS}+e_{RX}+e_{DPU})\Delta T,
\end{aligned}$$

Note that the nodes in regions Ω_2 and Ω_3 do not broadcast any information since they do not detect the target; however they can still receive the information from broadcasting nodes in their neighborhood.

4.5.1 Energy Consumption and Lifetime Characteristics

Theorem 4.5.1 *The expected energy consumption of the POSE.3C network during a ΔT time interval is given as*

$$\begin{aligned}\overline{E}_{\Delta T} &= N_{sel} m \overline{E}^{\Omega_1^*} + \left(\rho A_{\Omega_1} - N_{sel} m\right) \overline{E}^{\Omega_1^{*'}} + \\ &\quad \rho A_{\Omega_2} \overline{E}^{\Omega_2} + \rho A_{\Omega_3} \overline{E}^{\Omega_3}\end{aligned}$$

where ρ is the sensor network density; $\overline{E}^{\Omega_1^*}$, $\overline{E}^{\Omega_1^{*'}}$, $\overline{E}^{\Omega_2}$, and $\overline{E}^{\Omega_3}$ are the expected energy consumptions in regions $\Omega_1^*, \Omega_1^{*'}, \Omega_2$, and Ω_3, respectively, given as

$$\begin{aligned}\overline{E}^{\Omega_1^*} &= E_{LPS}^{\Omega_1} p_2^{\Omega_1^*} + E_{HPS}^{\Omega_1} p_3^{\Omega_1^*}, \\ \overline{E}^{\Omega_1^{*'}} &= E_{Sleep} p_1^{\Omega_1^{*'}} + E_{LPS}^{\Omega_1} p_2^{\Omega_1^{*'}}, \\ \overline{E}^{\Omega_2} &= E_{Sleep} p_1^{\Omega_2} + E_{LPS}^{\Omega_2,\Omega_3} p_2^{\Omega_2}, \\ \overline{E}^{\Omega_3} &= E_{Sleep} p_1^{\Omega_3} + E_{LPS}^{\Omega_2,\Omega_3} p_2^{\Omega_3} + E_{HPS}^{\Omega_2,\Omega_3} p_3^{\Omega_3},\end{aligned}$$

$[p_1, p_2, p_3]$ *are the steady state probabilities that a nose is in the* Sleep, LPS, *and* HPS *states within each region, respectively, and are given as follows:*

$$\begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix}^{\Omega_1^*} = \begin{bmatrix} 0 \\ 1-\alpha \\ \alpha \end{bmatrix}, \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix}^{\Omega_1^{*'}} = \begin{bmatrix} \frac{1-\alpha}{2-p_{sleep}-\alpha} \\ \frac{1-p_{sleep}}{2-p_{sleep}-\alpha} \\ 0 \end{bmatrix},$$

$$\begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix}^{\Omega_2} = \begin{bmatrix} \frac{1-p_{fa}}{2-p_{sleep}-p_{fa}} \\ \frac{1-p_{sleep}}{2-p_{sleep}-p_{fa}} \\ 0 \end{bmatrix}, \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix}^{\Omega_3} = \begin{bmatrix} \frac{(1-2p_{fa})}{2-p_{sleep}-2p_{fa}} \\ \frac{(1-p_{fa})(1-p_{sleep})}{2-p_{sleep}-2p_{fa}} \\ \frac{p_{fa}(1-p_{sleep})}{2-p_{sleep}-2p_{fa}} \end{bmatrix}$$

Proof: The state transition probability matrices for nodes in Ω_1^*, $\Omega_1^{*'}$, Ω_2, and Ω_3 are given as

$$\begin{aligned} P^{\Omega_1^*} &\triangleq \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1-\alpha & \alpha \\ 0 & 1-\alpha & \alpha \end{bmatrix}, \\ P^{\Omega_1^{*'}} &\triangleq \begin{bmatrix} p_{sleep} & 1-p_{sleep} & 0 \\ 1-\alpha & \alpha & 0 \\ 1-\alpha & \alpha & 0 \end{bmatrix}, \\ P^{\Omega_2} &\triangleq \begin{bmatrix} p_{sleep} & 1-p_{sleep} & 0 \\ 1-p_{fa} & p_{fa} & 0 \\ 1-p_{fa} & p_{fa} & 0 \end{bmatrix}, \\ P^{\Omega_3} &\triangleq \begin{bmatrix} p_{sleep} & 1-p_{sleep} & 0 \\ 1-p_{fa} & 0 & p_{fa} \\ 0 & 1-p_{fa} & p_{fa} \end{bmatrix}, \end{aligned} \tag{4.13}$$

where the operator $P^{\Omega_1^*}$ is for a selected node, i.e. if $s_i \in \mathcal{S}^*$, while $P^{\Omega_1^{*'}}$ is for a node that is not selected, i.e. if $s_i \notin \mathcal{S}^*$. Note that α is chosen as the best case conservative estimate of $P_{HPS}^{s_i}$. Similar to $P^{\Omega_1^{*'}}$, P^{Ω_2} is generated by putting $P_{D,LPS}^{s_i}(\mathbf{u}^{\tau_\ell}(k)) = p_{fa}$; while P^{Ω_3} is constructed for $\mathcal{N}_{HPS}^{s_i} = \emptyset$.

Based on the above operators, the steady-state probabilities of each state within each region are computed as follows

$$\begin{aligned} \left[\, p_1, p_2, p_3 \,\right] &= \left[\, p_1, p_2, p_3 \,\right] P \\ s.t.\ \ p_1 + p_2 + p_3 &= 1 \end{aligned} \tag{4.14}$$

For $P^{\Omega_1^*}$:

$$\begin{aligned}
p_1^{\Omega_1^*} &= 0, \text{ since a sleeping sensor cant be selected} \\
p_2^{\Omega_1^*} &= (1-\alpha)(p_2^{\Omega_1^*} + p_3^{\Omega_1^*}), \\
p_3^{\Omega_1^*} &= \alpha(p_2^{\Omega_1^*} + p_3^{\Omega_1^*}),\ p_3^{\Omega_1^*} = 1 - p_2^{\Omega_1^*}, \\
\Rightarrow &\qquad p_2^{\Omega_1^*} = 1-\alpha,\ p_3^{\Omega_1^*} = \alpha. \qquad (4.15)
\end{aligned}$$

For $P^{\Omega_1^{*'}}$:

$$\begin{aligned}
p_1^{\Omega_1^{*'}} &= p_{sleep}p_1^{\Omega_1^{*'}} + (1-\alpha)(p_2^{\Omega_1^{*'}} + p_3^{\Omega_1^{*'}}), \\
p_2^{\Omega_1^{*'}} &= (1-p_{sleep})p_1^{\Omega_1^{*'}} + \alpha(p_2^{\Omega_1^{*'}} + p_3^{\Omega_1^{*'}}), \\
p_3^{\Omega_1^{*'}} &= 0, \quad p_1^{\Omega_1^{*'}} = 1 - p_2^{\Omega_1^{*'}}, \\
\Rightarrow &\qquad p_1^{\Omega_1^{*'}} = \frac{1-\alpha}{2-p_{sleep}-\alpha}, \text{ and} \\
&\qquad p_2^{\Omega_1^{*'}} = \frac{1-p_{sleep}}{2-p_{sleep}-\alpha}. \qquad (4.16)
\end{aligned}$$

For P^{Ω_2}: By replacing α with p_{fa} in Eq. (4.16), we find the steady-state probabilities as $p_1^{\Omega_2} = \frac{1-p_{fa}}{2-p_{sleep}-p_{fa}}$, $p_2^{\Omega_2} = \frac{1-p_{sleep}}{2-p_{sleep}-p_{fa}}$, and $p_3^{\Omega_2} = 0$.

For P^{Ω_3}:

$$\begin{aligned}
p_1^{\Omega_3} &= p_{sleep}p_1^{\Omega_3} + (1-p_{fa})p_2^{\Omega_3}, \\
p_2^{\Omega_3} &= (1-p_{sleep})p_1^{\Omega_3} + (1-p_{fa})p_3^{\Omega_3}, \\
p_3^{\Omega_3} &= p_{fa}(p_2^{\Omega_3} + p_3^{\Omega_3}), \quad p_1^{\Omega_3} = 1 - p_2^{\Omega_3} - \frac{p_{fa}p_2^{\Omega_3}}{1-p_{fa}},
\end{aligned}$$

Solving the above and assuming $p_{fa}^2 << 1$, we get

$$\begin{aligned}
p_1^{\Omega_3} &= \frac{1-2p_{fa}}{2-p_{sleep}-2p_{fa}}, \\
p_2^{\Omega_3} &= \frac{(1-p_{fa})(1-p_{sleep})}{2-p_{sleep}-2p_{fa}}, \\
p_3^{\Omega_3} &= \frac{p_{fa}(1-p_{sleep})}{2-p_{sleep}-2p_{fa}}.
\end{aligned} \tag{4.17}$$

Next, we compute the expected number of sensor nodes in each state per region. Since a total of $N_{sel}m$ sensor nodes are selected around m targets in region Ω_1^* we get:

$$\overline{N}_{Sleep}^{\Omega_1} = (\rho A_{\Omega_1} - N_{sel}m)p_1^{\Omega_1^{*'}};\ \overline{N}_{Sleep}^{\Omega_2} = \rho A_{\Omega_2} p_1^{\Omega_2};$$

$$\overline{N}_{Sleep}^{\Omega_3} = \rho A_{\Omega_3} p_1^{\Omega_3};\ \overline{N}_{LPS}^{\Omega_1} = N_{sel}mp_2^{\Omega_1^*} + (\rho A_{\Omega_1} - N_{sel}m)p_2^{\Omega_1^{*'}}$$

$$\overline{N}_{LPS}^{\Omega_2} = \rho A_{\Omega_2} p_2^{\Omega_2};\ \overline{N}_{LPS}^{\Omega_3} = \rho A_{\Omega_3} p_2^{\Omega_3}$$

$$\overline{N}_{HPS}^{\Omega_1} = N_{sel}mp_3^{\Omega_1,*};\ \overline{N}_{HPS}^{\Omega_2} = 0;\ \overline{N}_{HPS}^{\Omega_3} = \rho A_{\Omega_3} p_3^{\Omega_3}. \tag{4.18}$$

Now, the average energy consumed by the network is

$$\begin{aligned}
\overline{E}_{\Delta T} = E_{Sleep}\overline{N}_{Sleep}^{\Omega_1} + E_{LPS}^{\Omega_1}\overline{N}_{LPS}^{\Omega_1} + E_{HPS}^{\Omega_1}\overline{N}_{HPS}^{\Omega_1} + \\
E_{Sleep}\overline{N}_{Sleep}^{\Omega_2} + E_{LPS}^{\Omega_2,\Omega_3}\overline{N}_{LPS}^{\Omega_2} + E_{HPS}^{\Omega_2,\Omega_3}\overline{N}_{HPS}^{\Omega_2} + \\
E_{Sleep}\overline{N}_{Sleep}^{\Omega_3} + E_{LPS}^{\Omega_2,\Omega_3}\overline{N}_{LPS}^{\Omega_3} + E_{HPS}^{\Omega_2,\Omega_3}\overline{N}_{HPS}^{\Omega_3}
\end{aligned} \tag{4.19}$$

Thus, putting the number of sensors derived above into Eq. (4.19), we obtain the result of the theorem. □

Definition 4.5.1 (Network Lifetime) *Consider a path γ of length L in the region Ω that is taken by the maximum number of targets. Now consider a cylindrical tube $\Omega_\gamma \subset \Omega$ of radius R_{HPS} around*

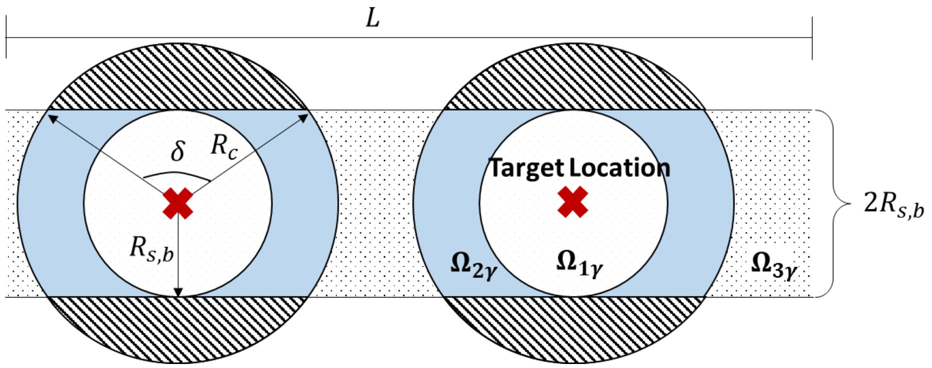


Figure 4.6: Illustration of the tube Ω_γ with 2 targets.

this path, which consists of sensors $\mathcal{S}_\gamma \subset \mathcal{S}$ that will die earliest in the network. The expected network lifetime, $\overline{T}_{Life}$, is then defined as the time when the energy of sensor nodes within Ω_γ reduces to a certain fraction $\eta \in [0, 1)$, s.t.

$$\frac{\sum_{s_j \in \mathcal{S}_\gamma}(E_0 - E^{s_j}(\overline{T}_{Life}))}{\sum_{s_j \in \mathcal{S}_\gamma} E_0} = \eta$$

Theorem 4.5.2 *The expected lifetime of a POSE.3C network is*

$$\overline{T}_{Life} = \frac{2\rho R_{HPS} L E_0 \Delta T(1-\eta)}{\overline{E}_{\Delta T}}$$

Proof: Let the number of TOIs or (TNOIs) traveling through Ω_γ per ΔT time interval be represented by a Poisson process given by

$$P(N_\gamma = m_\gamma) = e^{-\lambda \Delta_T} \frac{(\lambda \Delta_T)^{m_\gamma}}{m_\gamma!}. \tag{4.20}$$

Then, the average number of targets in Ω_γ is $\overline{N}_\gamma = \lambda$. Let the average velocity of a target is $\overline{V}$. Also, let L be the tube length which the target travels. Then the expected number of time intervals that a target spends in Ω_γ is $\overline{T} = L/\overline{V}\Delta T$.

Let $\overline{T}_{Life}$ be the expected life of the network. Now, consider multiple TOIs (or TNOIs) travelling in the tube Ω_γ. These targets lead to a partition of the tube into three regions, $\Omega_{1\gamma}$, $\Omega_{2\gamma}$, and $\Omega_{3\gamma}$, as shown in Fig. 4.6. Note that these regions are similar in characteristics to the regions Ω_1, Ω_2, and Ω_3, respectively, except that these are defined within the tube Ω_γ. To be conservative, we assume that the targets have disjoint detection and communication regions. Therefore, the expected energy consumption of the sensor nodes in ΔT time interval is as given in Theorem 4.5.1, where the areas therein are replaced by the corresponding areas $A^{\Omega_{1\gamma}}$, $A^{\Omega_{2\gamma}}$, and $A^{\Omega_{3\gamma}}$. Here, $A^{\Omega_{1\gamma}} = \lambda\pi R^2_{HPS}$; $A^{\Omega_{2\gamma}} = \lambda\left(\pi(R_c^2 - R^2_{HPS}) - R_c^2(\delta - sin\delta)\right)$, is the sector area as shown in Fig. 4.6, where $\delta = \left(\pi - 2tan^{-1}\left(\frac{R_{HPS}}{\sqrt{R_c^2 - R^2_{HPS}}}\right)\right)$; and $A^{\Omega_{3\gamma}} = 2LR_{HPS} - (A^{\Omega_{1\gamma}} + A^{\Omega_{2\gamma}})$. Now, for the life time intervals the total energy consumed is given as $\overline{E}_{\Delta T}\frac{\overline{T}_{Life}}{\Delta T}$.

Using Defn. 4.5.1, we can solve for $\overline{T}_{Life}$ as follows:

$$\frac{\sum_{s_j \in \mathcal{S}_\gamma}(E_0 - E^{s_j}(\overline{T}_{Life}))}{\sum_{s_j \in \mathcal{S}_\gamma} E_0} = \eta \tag{4.21}$$

$$\Rightarrow 1 - \frac{\overline{E}_{\Delta T}\overline{T}_{Life}}{\Delta T \rho 2 R_{HPS} L E_0} = \eta \tag{4.22}$$

$$\Rightarrow \overline{T}_{Life} = \frac{2\rho R_{HPS} L E_0 \Delta T(1-\eta)}{\overline{E}_{\Delta T}} \tag{4.23}$$

□

4.5.2 Missed Detection Characteristics

Definition 4.5.2 (Target Birth) *A target birth is the time instance when a target first appears in the deployment region Ω.*

Definition 4.5.3 (Mature Target) *A mature target is a target that has travelled inside the region Ω for sufficient time such that sensor collaboration has taken place to track it.*

Theorem 4.5.3 *The missed detection probability characteristics of a POSE.3C network are given as follows:*

a) For a target birth:

$$P_{m,bir} \geq exp\left(-\frac{\pi R_r^2 \alpha \chi \rho (1 - p_{sleep})}{2 - p_{sleep} - 2p_{fa}}\right)$$

b) For a mature target:

$$P_{m,mat} \geq exp\left(-\frac{\pi R_r^2 \alpha\chi\rho\left[(1-p_{sleep})+\frac{N_{sel}}{\rho\pi R_{HPS}^2}(1-\alpha)\right]}{2-p_{sleep}-\alpha}\right)$$

where $\chi = 1+\frac{2(1+\beta R_r)}{\beta^2 R_r^2}\left(1-\frac{(1+\beta R_{HPS})e^{-\beta R_{HPS}}}{(1+\beta R_r)e^{-\beta R_r}}\right)$

Proof: The probability q that an individual sensor s_i sampled from the deployment region Ω detects the target, is represented by a spatial Poisson process [47], such that

$$q \;= Pr\left\{Det_i = 1\right\} = \;1-e^{-\varphi(p_2+p_3)}, \tag{4.24}$$

where $Det_i = 1$ denotes a detection, and φ is the coverage factor for the sensor which is computed as:

$$\varphi = \frac{1}{A_\Omega}\left(\int_0^{R_r} 2\pi r\alpha dr + \int_{R_r}^{R_{HPS}} 2\pi r\alpha e^{-\beta(r-R_r)}dr\right) \tag{4.25}$$

Using integration by parts and simplifying we get:

$$\varphi = \frac{\pi R_r^2\alpha}{A_\Omega}\left[1+\frac{2(1+\beta R_r)}{\beta^2 R_r^2}\left(1-\frac{(1+\beta R_{HPS})e^{-\beta R_{HPS}}}{(1+\beta R_r)e^{-\beta R_r}}\right)\right] \tag{4.26}$$

Since the term within the square brackets in Eq. (4.26) is equal to χ, $\varphi = \frac{\pi R_r^2\alpha\chi}{A_\Omega}$. Here p_2+p_3 is the probability that the sensor is in the *LPS* or *HPS* state, and thus capable of detecting the target. Since each sensor is statistically independent and identical, the probability of exactly κ sensor detections [47] is given by Bernoulli trials as follows

$$Pr\left\{\sum Det_i = \kappa\right\} \;=\; \binom{n}{\kappa} q^\kappa(1-q)^{n-\kappa} \tag{4.27}$$

Then the probability of missed detections is given as

$$P_m = Pr\left\{\sum Det_i = 0\right\} = \binom{n}{0} q^0(1-q)^n = e^{-\varphi(p_2+p_3)n} \tag{4.28}$$

a) For a target birth: Since a target has just taken birth, the entire deployment region follows the state transition probabilities corresponding to the matrix P^{Ω_3}, as given in Eq. (4.13). Also, $n = \rho A_\Omega$. Substituting for φ, n, and $p_2^{\Omega_3} + p_3^{\Omega_3}$ in Eq. (4.28), the lower bound on P_m for a target birth is given as

$$P_{m,bir} \geq exp\left(-\frac{\pi R_r^2 \alpha\chi\rho(1-p_{sleep})}{2-p_{sleep}-2p_{fa}}\right)$$

b) For a mature target: Since the target is mature, sensor collaborations are taking place to select the optimal sensors for target tracking. Thus, the chosen sensor could be a selected sensor or a not selected sensor. Therefore, it follows the state transition probabilities corresponding to the matrix $P^{\Omega_1^\star}$ or $P^{\Omega_1^{*'}}$, respectively. Thus the probability $p_2 + p_3$ to find a sensor in the LPS or HPS state is given as

$$p_2 + p_3 = \frac{N_{sel}}{\rho\pi R_{HPS}^2}(p_2^{\Omega_1^\star} + p_3^{\Omega_1^\star}) + \frac{\rho\pi R_{HPS}^2 - N_{sel}}{\rho\pi R_{HPS}^2}(p_2^{\Omega_1^{*'}} + p_3^{\Omega_1^{*'}})$$

$$\Rightarrow p_2 + p_3 = \frac{N_{sel}}{\rho\pi R_{HPS}^2} + \frac{\rho\pi R_{HPS}^2 - N_{sel}}{\rho\pi R_{HPS}^2}\left(\frac{1-p_{sleep}}{2-p_{sleep}-\alpha}\right)$$

$$\Rightarrow p_2 + p_3 = \frac{N_{sel}(1-\alpha) + \rho\pi R_{HPS}^2\,(1-p_{sleep})}{\rho\pi R_{HPS}^2(2-p_{sleep}-\alpha)} \tag{4.29}$$

Substituting for φ, n, and $p_2 + p_3$ in Eq. (4.28), the lower bound on P_m for a mature target is given as

$$P_{m,mat} \geq exp\left(-\frac{\pi R_r^2\alpha\chi\rho\left[(1-p_{sleep}) + \frac{N_{sel}}{\rho\pi R_{HPS}^2}(1-\alpha)\right]}{2-p_{sleep}-\alpha}\right)$$

□

4.5.3 Theorem Validations

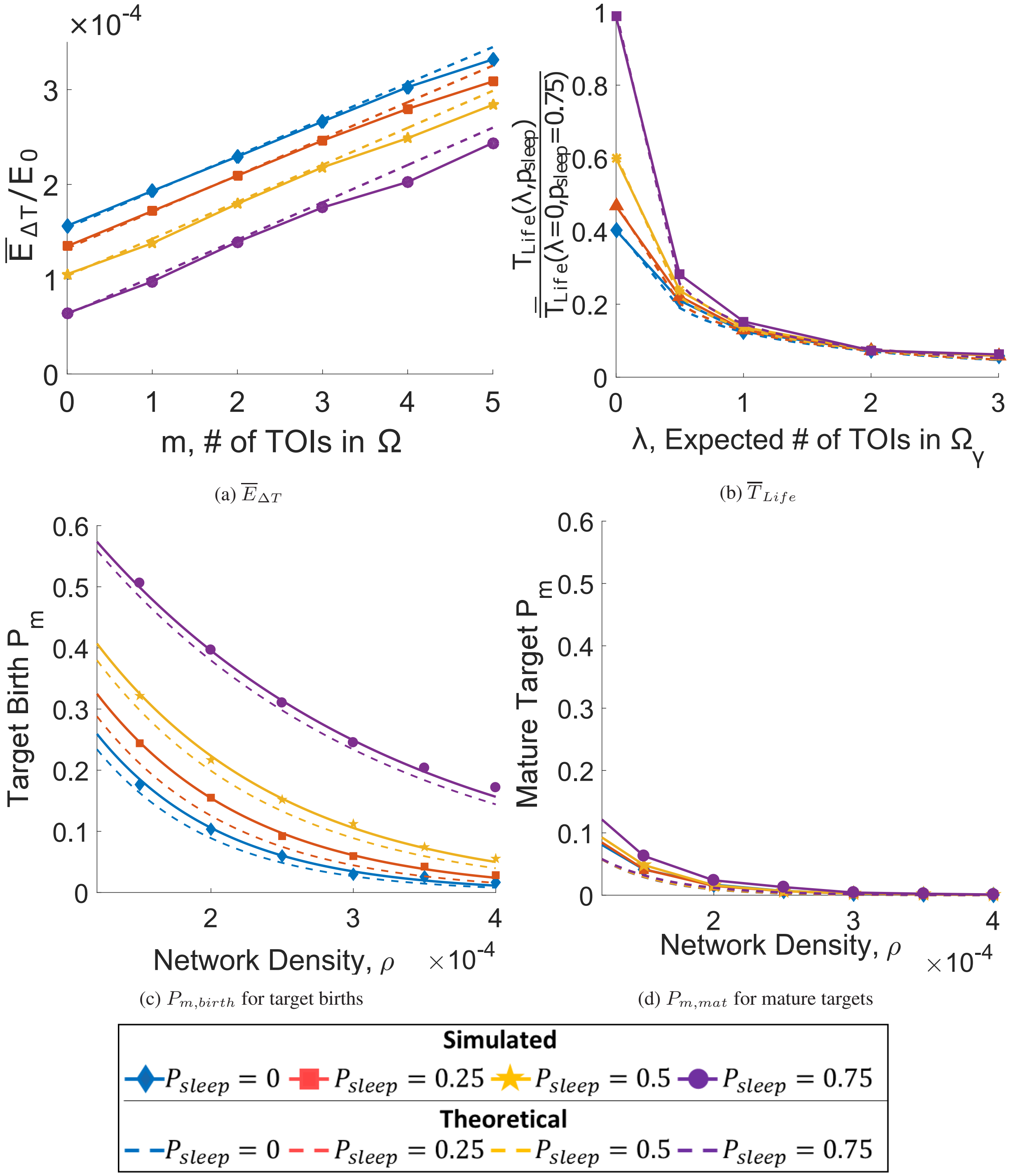
In order to validate the theorems, the POSE.3C algorithm was simulated in a 1km x 1km deployment region. For a thorough analysis, 500 Monte-Carlo simulation runs were conducted where the distribution of sensor nodes was regenerated during each run according to a uniform distribution. Table 5.1 lists the different simulation parameters.

To validate Theorems 4.5.1 and 4.5.2, the network characteristics in terms of expected energy consumption and lifetime, were evaluated against the number of targets present and the design parameter p_{sleep}. The network density was chosen to be $\rho = 4e^{-4}\ \frac{nodes}{m^2}$. For Theorem 4.5.1, m targets were deployed throughout Ω according to a uniform distribution. Subsequently, the average network energy consumption was acquired over the different simulation runs. Fig. 4.7a shows the comparison of the simulated vs theoretical results of the average energy consumption for different number of targets and for various p_{sleep} values. For Theorem 4.5.2, a tube $\Omega_\gamma \subset \Omega$ was considered of length $L = 600m$ and width $2R_{HPS} = 180m$. The targets were simulated with an arrival rate such that an average of λ targets are traveling through the tube during each time interval. The simulations were run until the network reached its lifetime according to Defn. 4.5.1. In this manner the network lifetime $\overline{T}_{Life}(\lambda, p_{sleep})$ was computed for different values of λ and p_{sleep}. The lifetime is normalized by the expected lifetime for $\lambda = 0$ and $p_{sleep} = 0.75$. Fig. 4.7b shows the comparison of the simulation and theoretical results for the expected lifetime. As seen in both Figs. 4.7a and 4.7b, the simulation results match the theoretical results. It is observed that as we increase p_{sleep}, the expected energy consumption decreases while the expected lifetime increases.

To validate Theorem 4.5.3 Part a, a random target birth was generated in the region Ω at each time instance, while to validate Part b, a moving target was generated traveling through the region

Table 4.2: Simulation Parameters

$e_{clock} = 0.01W$	$R_r = 60m$	$b_{11} = b_{22} = 0.9$
$e_{LPS} = 2.5mW$	$R_{HPS} = R_{LPS} = 90m$	$b_{12} = b_{21} = 0.1$
$e_{HPS} = 22W$	$R_c = 180m$	$N'_{sel} = 5$
$e_{TX} = 1.26W$	$L = 600m$	$\sigma_\phi = 1°$
$e_{RX} = 0.63W$	$\alpha = 0.95$	$\sigma_R = 1m$
$e_{DPU} = 1W$	$\beta = 0.00171$	$\sigma_{\boldsymbol{v},x} = \sigma_{\boldsymbol{v},y} = 1m$
$E_0 = 1.08MJ$	$\eta = 0.05$	$\sigma_{\boldsymbol{v},\psi} = 1°$
$\Delta T = 0.5s$	$p_{fa} = 0.01$	$\mu_{cl} = 0.025$



(a) $\overline{E}_{\Delta T}$

(b) $\overline{T}_{Life}$

(c) $P_{m,birth}$ for target births

(d) $P_{m,mat}$ for mature targets

Figure 4.7: Validation of: (a) Theorem 4.5.1, (b) Theorem 4.5.2, (c) Theorem 4.5.3 Part a, and (d) Theorem 4.5.3 Part b.

according to the DWNA model. The probabilities of missed detection for target births and mature targets were computed over the Monte Carlo runs by counting the number of detections and misses. The simulations were repeated for various network densities and p_{sleep} values. The probabilities

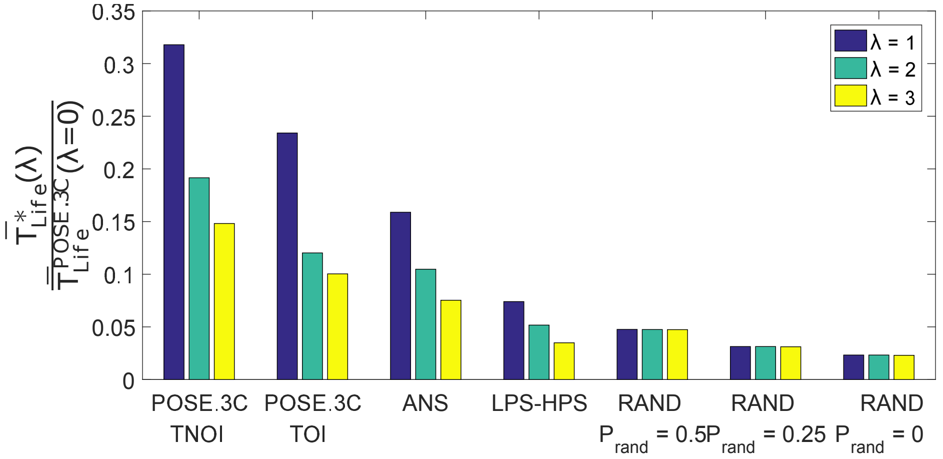


Figure 4.8: Network Lifetime Comparison

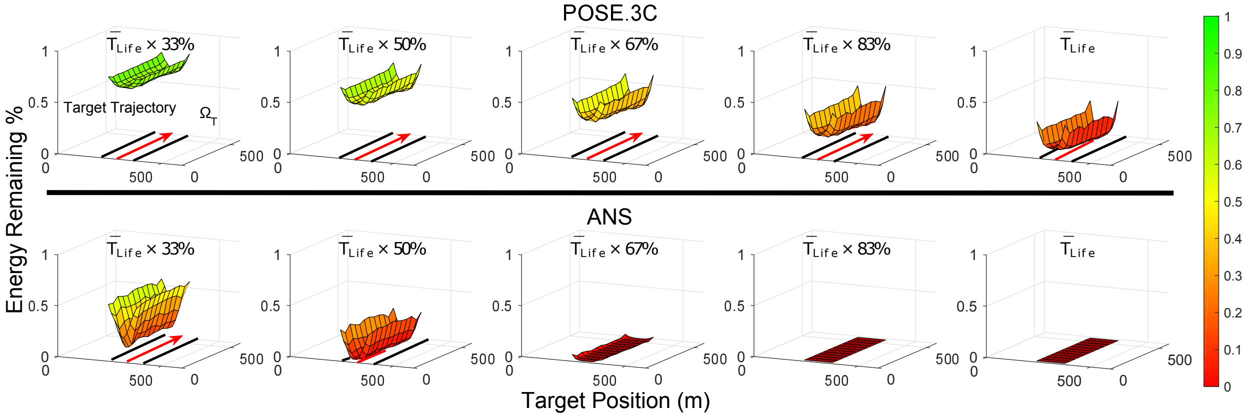


Figure 4.9: Distribution of Energy Remaining around a TOI

are plotted in Figs. 4.7c and 4.7d, which show that the simulation results match the theory. It is also seen that $P_{m,mat}$ is significantly lower than $P_{m,birth}$. This is because sensor collaborations allow the network to activate their LPS and HPS devices in advance to detect and track the target.

4.6 POSE.3C Algorithm Performance

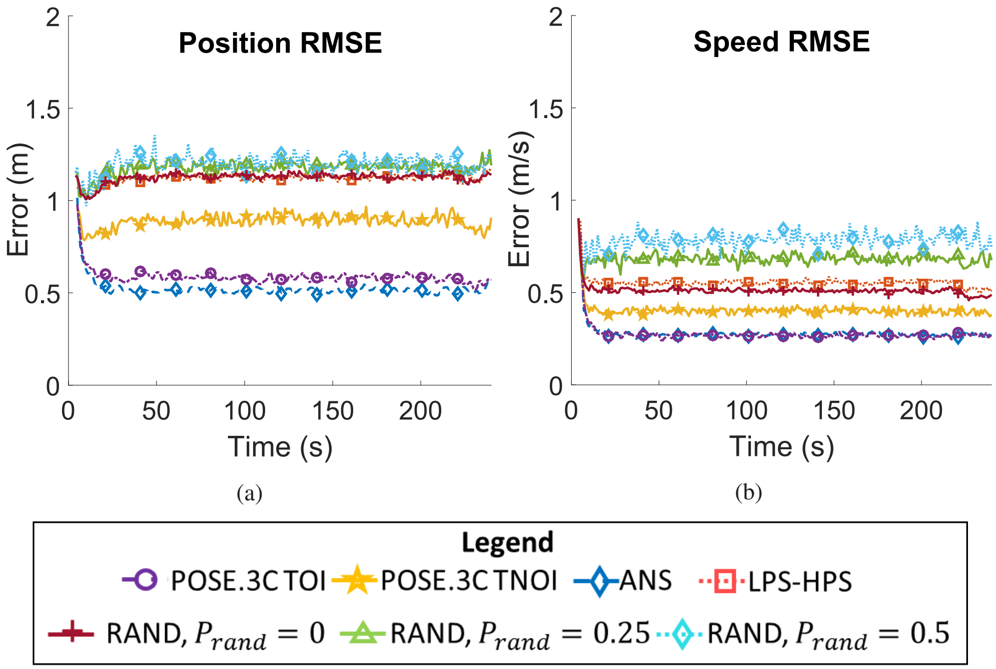
To verify that the proposed network meets thesis objective, this section presents the results of the POSE.3C algorithm compared with existing methods. Specifically, POSE.3C is compared with the following distributed scheduling methods: (1) *Autonomous Node Selection* (ANS), (2) *LPS-HPS Scheduling*, and (3) *Random Scheduling*. The ANS algorithm [29] is a distributed sensor selection method that utilizes a cost function that minimizes the Mean Square Error (MSE) based on the GDOP. Here, the sensors collaborate in a distributed manner to make the scheduling decision;

however the sensor states are only binary, i.e. passive (*LPS*) and active (*HPS*). Thus, if a sensor node is selected, it switches to the active state, while if it is not selected, it remains in the passive state. The LPS-HPS Scheduling method is a distributed trigger-based activation method where the sensor nodes remain in the passive (*LPS*) state until a target is detected. Once a node detects a target, it switches to the active (*HPS*) state. The Random Scheduling method is a distributed method where the sensor nodes probabilistically cycle between actively sensing (*HPS*) and sleeping. Thus, during each time step the nodes sleep with a probability P_{rand}, while they sense the environment in the *HPS* state with a probability $1 - P_{rand}$. Thus, for $P_{rand} = 0$, the sensor nodes are always sensing, while for $P_{rand} = 1$ they are always sleeping. Note that in both the LPS-HPS and Random Scheduling methods, the sensor nodes do not collaborate.

The sensor nodes are assumed to have a hydrophone array [81] as the LPS device and an active sonar [82] as the HPS device. Table 5.1 lists the different simulation parameters including energy costs, sensing parameters, process noise parameters ($\sigma_{\boldsymbol{v},x}, \sigma_{\boldsymbol{v},y}, \sigma_{\boldsymbol{v},\psi}$), and measurement noise parameters ($\sigma_{\phi}, \sigma_{R}$). The network density, $\rho = 4e^{-4}$, and design parameter, $p_{sleep} = 0.5$, are chosen to ensure that the probability of missed detection for a mature target is less than 0.01. The number of sensor nodes selected to track a TOI is chosen to be $N_{sel} = 3$, while for a TNOI $N_{sel} = 1$.

First, the network life expectancy of the POSE.3C and the other methods is computed by varying the expected number of targets, λ, that travel within a tube $\Omega_{\gamma} \in \Omega$. The results achieved are presented in Fig. 4.8 where each bar is normalized by the life expectancy of the POSE.3C network for $\lambda = 0$. As seen, the life expectancy of the POSE.3C network is higher than all of the other scheduling methods. Also, as expected the lifetime achieved by the POSE.3C network while tracking TNOIs is higher than that for TOIs. Additionally, it can be concluded that if there is a mix of TOIs and TNOIs within the network, then the life expectancy of the POSE.3C network will lie between the POSE.3C TNOI and TOI values.

Next, the distribution of energy remaining around the targets within a tube $\Omega_{\gamma} \in \Omega$ of the POSE.3C network is compared with that of the ANS algorithm. For this comparison, the expected number of targets within the tube is chosen as $\lambda = 1$. Fig. 4.9 shows several snapshots of the remaining energy distribution within Ω_{γ} until $\overline{T}_{Life}$ of the POSE.3C is reached. As seen, the



(a)

(b)

Figure 4.10: Root Mean Squared Error of (a) Position Estimate and (b) Speed Estimate

distribution of energy within the tube is much more uniform for the POSE.3C network as compared to that of the ANS network. This is because the ANS network depletes the energy rapidly near the track by always selecting the closest sensors to minimize the tracking error, while the POSE.3C network allows for energy based ranking while maintaining tracking accuracy. A slight increase in energy on the sides of the tube is seen due to boundary effects. Also, the ANS network dies much more rapidly as compared to the POSE.3C network.

Additionally, the tracking accuracy is evaluated for each network according to the Root Mean Squared Error (RMSE) and is presented in Fig. 4.10. As seen in the figure, the POSE.3C and ANS networks have very low tracking errors as compared to the Random and LPS-HPS methods due to distributed fusion and collaboration between neighbors.

The complexity of the POSE.3C algorithm arises from the JPDA, Distributed Fusion, and Distributed Sensor Selection algorithms within the HPS and LPS states. The computational complexity was analyzed by measuring the average times taken for each of these processes and the results are shown in Table 4.3. These average times were generated in a Matlab environment on a i5 3.1 GHz CPU computer.

Table 4.3: Computation Times of the POSE.3C Components

	JPDA	Distributed Fusion	Distributed Sensor Selection
TOI	1.4 ms	2.6 ms	0.37 ms
TNOI	1.3 ms	2.37 ms	0.36 ms

4.7 Conclusion and Limitations

This chapter developed the POSE.3C algorithm for controlling a distributed sensor network for energy-efficient target tracking. The underlying network control approach consists of detecting, classifying, and predicting the targets location as they travel through the deployment region. The information obtained is then used to classify the type of target to allow the network to opportunistically cluster *HPS* nodes that are the most reliable and optimal sensor nodes around targets of interest. The multi-modal states of each sensor node are probabilistically controlled using distributed PFSA-based supervisors. Theoretical properties of the POSE.3C network characteristics have been established and validated to show that the POSE.3C network significantly increases the network lifetime while providing high tracking accuracy and a low probability of missed detection.

However, there are two limitations that were not addressed in this chapter. First, this approach does not discuss the issues that arise when multiple co-located sensor nodes fail. Although the POSE.3C algorithm is robust to single sensor node failures, if multiple co-located sensor nodes fail, a coverage gap is formed. If the target travels through the coverage gap, then the POSE.3C network will miss the target. Additionally, this approach does not discuss how to ensure target coverage when the network density is non-uniform. Since the sensor nodes may be air dropped or thrown into the ROI, the nodes may end up clustering together to form high and low density regions. If a target travels into a low density region, the network must be able to adapt to ensure target coverage. Therefore, the next chapter presents the *Prediction-based Opportunistic Sensing for Resilience* (POSE.R) algorithm to address the above issues.

CHAPTER 5

PREDICTION-BASED OPPORTUNISTIC SENSING FOR RESILIENCE (POSE.R)

5.1 Resilient Target Coverage Introduction

A critical challenge in *Distributed Sensor Networks* (DSN) that perform various intelligence, surveillance, and reconnaissance (ISR) operations, is to maintain *Target Coverage* in the event of sensor node failures. Sensor nodes are prone to failures due to component degradations, hardware failures, malicious attacks, or battery depletions [33] causing changes in the network topology. If multiple co-located sensors fail, a sector of the network may be uncovered, causing missed detections when a target travels through this coverage gap. This results in poor network performance, information delays, and mission failures. Additionally, the sensor nodes may be non-uniformly distributed causing high and low density regions and even coverage gaps due to environmental conditions and uncertainties [34]. Therefore, the development of a self-adaptive network that mitigates the formation of coverage gaps is necessary to ensure network resilience.

To account for node failures, two proactive approaches have been proposed in literature: (i) redundant node deployment and (ii) intelligent network control for energy-efficiency and life-extension. The former approach deploys redundant sensor nodes throughout the *Region Of Interest* (ROI) to ensure that every point is observed by $\kappa > 1$ nodes [37, 83]. This creates a fault-tolerant network that allows for $\kappa - 1$ nodes to fail before a coverage gap is formed; however, it is costly. Moreover, this approach does not provide resilience if multiple spatially co-located nodes fail, for example, an attack in a battlefield scenario.

The second proactive approach incorporates an intelligent network control strategy that minimizes node failures caused by energy depletion. One control strategy, known as *Opportunistic Sensing* [84], consists of selecting and activating sensor nodes only in the local regions around target predictions, while the nodes not selected are deactivated to conserve energy. This method allows for accurate state estimation of dynamic targets while maximizing the network lifetime. Another control strategy is achieved by adjusting the nodes sensing range [85]. This approach assumes that the node's sensing range can vary based on the amount of power supplied and aims to optimize the node's range and activation time to minimize energy consumption and missed detections. However, theses approaches assume that the target's are fixed and known *a priori* and do not consider tracking dynamic targets. Although intelligent control strategies allow the network to preserve energy, they only study energy-efficient control and do not address the problem of resiliency.

Therefore, this chapter proposes a distributed supervisory control algorithm, called *Prediction-based Opportunistic Sensing for Resilience* (POSE.R), for energy management and resiliency that adjusts the node's sensing range between $[R_1, R_L]$ based on predicted target locations. This algorithm extends the POSE and POSE.3C algorithms [84] by incorporating an adaptive distributed sensor selection approach that selects the best sensor nodes and their sensing ranges to track the target and ensure target coverage in the presence of failures and non-uniform node locations. The sensor selection approach adapts to the density of active sensor nodes around the target's predicted location as seen in Fig. 5.1. For high density regions with $\geq N_{sel}$ nodes, an Energy-based Geometric Dilution of Precision (EGDOP) sensor selection method is employed to select and activate geometrically diverse nodes with high remaining energy to cover the target with their minimum sensing range, R_1.

When a target is predicted to travel through a low density region, the sensor nodes may need to expand their sensing range to accommodate for a coverage gap or an insufficient number of nodes.

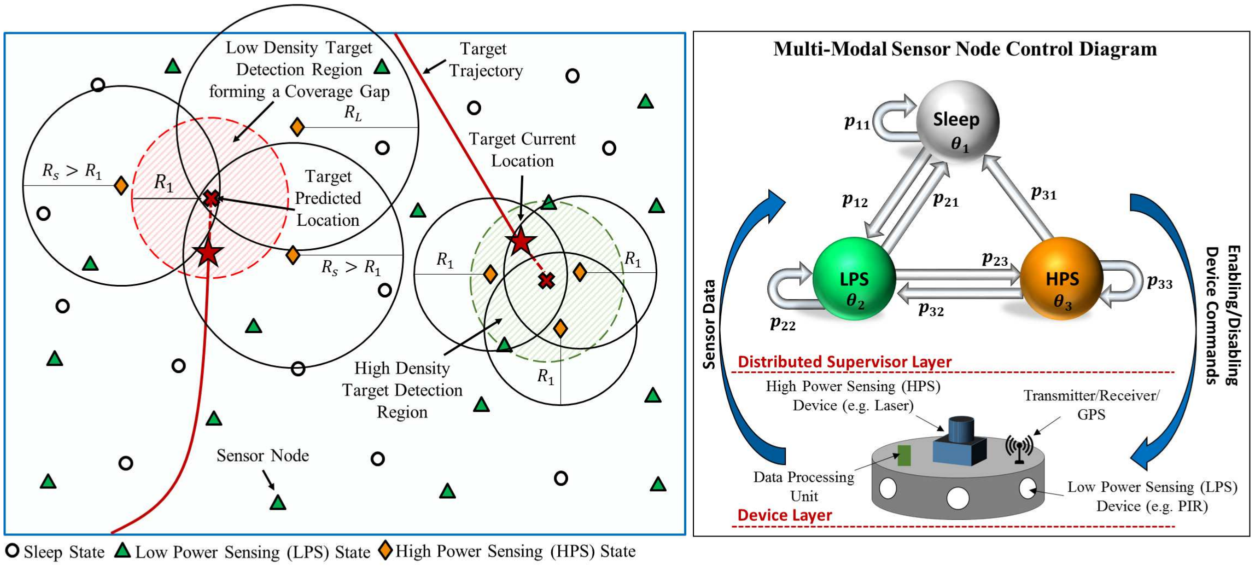

Low Density Target Detection Region forming a Coverage Gap
Target Trajectory
R_L
$R_s > R_1$
R_1
Target Predicted Location
Target Current Location
$R_s > R_1$
R_1
R_1
R_1
High Density Target Detection Region
Sensor Node
Multi-Modal Sensor Node Control Diagram
Sleep θ_1
LPS θ_2
HPS θ_3
p_{11}
p_{12}
p_{21}
p_{31}
p_{23}
p_{32}
p_{22}
p_{33}
Sensor Data
Enabling/Disabling Device Commands
Distributed Supervisor Layer
High Power Sensing (HPS) Device (e.g. Laser)
Transmitter/Receiver/ GPS
Data Processing Unit
Low Power Sensing (LPS) Device (e.g. PIR)
Device Layer
○ Sleep State ▲ Low Power Sensing (LPS) State ◆ High Power Sensing (HPS) State

Figure 5.1: Illustration of the POSE.R algorithm

To identify the best sensing range to cover the target while minimizing energy consumption, a Game Theoretic sensor selection method is employed using Potential Games. This approach to sensor selection provides the following advantages: (1) Non-cooperative games allow for scalable distributed computing necessary in DSN; (2) Potential games ensure that an equilibrium exists; and (3) maximizing the local objective function guarantees that the global objective is maximized. The objective function is select N_{sel} nodes and their sensing ranges to cover the target's predicted location and minimize energy consumption.

To manage the networks energy consumption, a *Probabilistic Finite State Automaton* (PFSA) is embedded on each sensor node to control the nodes multi-modal operating states by enabling/disabling its devices at each time step. The states of the PFSA include: 1) *Sleep*, 2) *LPS*, and 3) *HPS*. The *Sleep* state consumes minimal energy by disabling all devices. The *LPS* state utilize the LPS devices for target detection while conserving energy, and the *HPS* state enables the HPS devices for state estimation. The range of the HPS devices is varied between $[R_1, R_L]$ based on the adaptive sensor selection algorithm to ensure target coverage while minimizing redundant coverage and energy consumption. The transceiver is enabled in the *LPS* and *HPS* states for information sharing and collaboration.

The state transition probabilities of the PFSA are dynamically updated based on the adaptive sensor selection results and the information observed with the on-board sensing suite. The probabilities are designed to transition the node to the *HPS* state only when it is selected and a target is predicted to travel within it's coverage areas, while the node transitions between low power consuming states, *LPS* or *Sleep*, to conserve energy when not selected. This is illustrated in Fig. 5.1, where $N_{sel} = 3$ nodes are selected to be in the *HPS* state around the targets predicted locations to ensure target coverage, while the remaining sensors conserve energy. Even in low density regions that contain a coverage gap, the POSE.R network is able to adapt the node's sensing characteristics to fill the gap, thus adding resiliency while providing significant energy savings.

The main contributions of this chapter is the development of a distributed algorithm that facilitates resilient target coverage and tracking with the following attributes:

I distributed coverage gap identification method that does not rely on active or passive monitoring methods,

II distributed node selection approach that adapts to the network density around a target's predicted location via a) EGDOP selection criteria for high density regions, and b) a Game Theoretic node and range selection method using potential games for low density regions.

III distributed supervisory control strategy to enable/disable multi-modal sensing devices for energy-efficiency.

5.2 Related Work

The problem that this chapter aims to solve encompasses three main research topics discussed throughout literature. The first topic studied handles network control protocols when sensors have failed within the network. The next approach proposed is to schedule the sensor nodes to maximize the network lifetime while ensuring target coverage. The following subsections describe the state-of-the-art research and the limitations that will be addressed in this work.

5.2.1 Fault-tolerant WSN

Since sensor networks are deployed in harsh environments (such as battlefields), they are prone to attacks that can lead to sensor failures. This may result in a lose of coverage, connectivity, and in extreme cases, mission operation. The literature typically models the types of failures as either, (1) single node failure or (2) multiple node failure where the latter can compose of spatially independent single node failures or co-located single node failures [33]. A single node can fail due to energy depletion, hardware degradation, environmental hazards, etc, leading to a loss of data and small holes in the network. Whereas multiple nodes spatially co-located can fail due to malicious attacks, such as bombings, which can create large holes in the network requiring recovery mechanisms.

For both failure models, it is critical that the network detects and locates the failed nodes in order to enable fault recovery mechanisms. Fault detection methods proposed are typically achieved through active or passive monitoring. Active monitoring is achieved using a centralized or cluster-based network topology [35] and consists of requesting constant updates throughout the network or using heartbeat signals. Passive monitoring is achieved in centralized, cluster-based, or distributed network topologies, by observing the traffic already present in the network to infer the nodes health [36]. Since this paper focuses on a distributed network control strategy, passive monitoring methods are only considered.

One approach to detect faulty nodes is to assume that the healthy measurements are spatially correlated while faulty measurements are uncorrelated. This was achieved using a Bayesian [86] and Naymen-Pearson [54] formulation, where the likelihood of the sensor data was tested against neighboring data hypotheses. A variation of these approaches was presented by Ding et al. [87] where the median of the neighborhood measurements was used as a validation criteria for the sensor data. An approach to weighting the faulty measurements was presented by Sridhar et. al. [88] that does not eliminate faulty sensor data but gives it less weight during data fusion. When the measurements are uncorrelated the weight decreases while it increases when they are correlated. Based on sensor node health status, a Markov model for characterizing the networks reliability and Mean Time To Failure (MTTF) was presented by Munir et. al. [89]. Further understanding of the different fault detection methods can be found in [89]. Although these methods identify nodes that are providing faulty measurements, they cannot identify failed sensor nodes without probing the network. Therefore, the development of a passive fault detection method is necessary to identify coverage gaps caused by failed nodes.

To achieve passive fault detection in this work, a distributed filter-based sensor selection method is employed that selects the best healthy sensors to track the target. Sensor node, s_i, initially assumes that all of the sensors that can detect the target are asleep or have failed. Then, each sensor (including s_i) that is awake or healthy (that can detect the target) will transmit information to s_i and their neighbors. This allows each sensor to identify the healthy nodes in their neighborhood and identify if they are the best sensor to cover the target. Additionally, this allows for a passive method to identify coverage gaps based on the number of nodes available.

To handle faulty sensor data, three approaches are used in this work. First, false alarms are minimized by employing the $M - of - N$ track validation method [72]. Then, the target state information received by each sensor node is validated based on the target state estimation error and faulty (high error) information is disregarded [84]. Finally, the healthy (trustworthy) information received is then correlated to ensure accurate data fusion. This multi-stage process prevents faulty data from passing through the network and enhances the target state estimates.

Once the faulty nodes are detected, it is critical that the network performs a recovery mechanism. This can be broken down into two approaches, proactive and reactive. One proactive approach for a target tracking application proposes that transmitting target state information to regions along the track of the target leads to fault tolerant tracking [90]. This approach is utilized in this work by incorporating opportunistic sensing into the DSN. By transmitting target state information along the target path, the sensors can proactively prepare for the target allowing for fault-tolerance. However, these approaches are not fault-tolerant if a coverage gap is formed along the target's trajectory.

Therefore, most proactive approaches try to handle the problem during deployment by inserting redundant sensing nodes throughout the network. The number of nodes deployed are typically computed to achieve κ-coverage [37, 83, 91] or κ-connectivity [92–94] where κ is the number of sensors that can cover a point/target or is the number of communication paths. Other deployment topology methods utilize sensor placement models of four, six, and eight nodes that provide fault tolerant properties [95]. These network topologies allow for $\kappa - 1$ sensors to fail without losing coverage or connectivity. However, they require significantly more sensor nodes to be deployed, and in the event of multiple co-located failures, this approach will fail to ensure coverage.

Reactive approaches aim to recover coverage or connectivity that was lost due to the failed nodes. For a single and multiple spatially independent node failures, many approaches have been proposed that utilize mobile sensor nodes that can re-position themselves to ensure that connectivity is maintained. A summary of these approaches are presented in [33] and consist of re-positioning nodes based on minimizing the distance traveled, coverage loss, message overhead, recovery time, length of data paths, and number of relocated nodes. However, these approaches focus on a mixed network of mobile and stationary nodes, which are related to mobile sensor networks and not DSN.

For stationary sensor networks, single sensor failure recovery methods have been proposed. These include, storing redundant data for data recovery [39]; re-routing connectivity paths around the failed node or adjusting packet size sent to the failed node [40]; or re-configuring clusters to recover child nodes from a failed cluster head [21, 96–98]. However, these methods do not address the problem of recovering the coverage gap created by the failed node.

To recover the coverage gap, this chapter aims to utilize adjustable sensing ranges to expand the nodes coverage capabilities to ensure that the coverage gap is filled only when a target is traveling through it, thus allowing for *Opportunistic Coverage*. Additionally, the distributed filter-based sensor selection method aims to select κ nodes to cover the target even in the presences of coverage gaps.

When multiple co-located nodes fail in a stationary network, the network can be partitioned into multiple divisions. To reconnect the network, heuristic approaches have been proposed to find the optimal stationary relay node placement that will ensure that the partitions are connected for data aggregation [33, 38]. The stationary relay nodes are only used to transmit information and do not have any sensing capabilities. These approaches aim to minimize the number of stationary relays deployed, the quality of the topology, and to guarantee link bandwidth. Although these approaches ensure that the network can transmit data to a base station, they do not address the coverage problem and are mainly concerned with large scale failures.

The proposed control methodology in this paper aims to incorporate coverage resiliency into the DSN to overcome coverage gaps imposed by multiple co-located node failures. It will be shown that as long as the size of the coverage gap is $\leq R_L$, the nodes in the POSE.R network will opportunistically expand their sensing ranges when a target is traveling into the coverage gap to ensure target coverage at the expense of energy consumption. Although the connectivity problem is not addressed in this work, it is assumed that frequent fly-bys with UAV's could gather target information in the event of network partitioning.

5.2.2 Maximum Network Lifetime Problem

The second problem that the POSE.R network aims to solve is the *Maximum Network Lifetime with Adjustable Range* (MNLAR) problem for target coverage. The objective of the MNLAR problem is two fold: (1) scheduling the sensor nodes within the network for energy efficiency by activating and deactivating them periodically, and (2) selecting the active sensors and their adjustable sensing range to ensure that every target in the network is covered. More formally, the MNLAR problem is defined as:

Definition 5.2.1 (Maximum Network Lifetime with Adjustable Ranges Problem) *Given n sensor nodes and m targets, determine a set of sensor, their sensing range, and their activation time such that:*

1. *The network lifetime is maximized,*
2. *The set of active sensors covers all of the targets,*
3. *The sensor nodes cannot consume more than their initial energy, E_0, and*
4. *Each sensor can only select one sensing range at each time step.*

This problem has been formulated as an optimization problem in the form of Integer Programming [85, 99, 100], Linear Programming [101–103], and Voronoi Graphs [104, 105]. This problem is NP-complete [99] making it very difficult to perform in real-time. Therefore, many of the solutions presented are based on centralized or distributed heuristics.

For centralized heuristics, many approaches aim to identify the family of set covers that allow for all of the targets to be covered. Each set cover consists of a subset of sensor nodes that will be active and can over the targets. The goal is the maximize the number of cover sets and each nodes sensing range to maximize the network lifetime. The scheduling sequence only activates one set cover at a time while all other covers are inactive. This was solved using the *Adjustable Range Set Covers* (AR-SC) algorithm [99] which develops a Linear Programming heuristic to approximate the Integer Programming solution. The *Sensor Network Lifetime Problem* (SNLP) formulation [101] utilized the Garg-Konemann algorithm to approximate the optimal linear programming solution within a small factor, while the Column Generation algorithm was used by Cerulli et. al. [102]. The greedy heuristic proposed by Cerulli et. al. was adjusted by Mohamadi et. al. [106] by developing a learning automata-based algorithm to find the optimal cover sets. Additionally, the MLAR problem was extended to directional (such as camera) sensor networks [103].

For distributed heuristics, many approaches follow a greedy-based scheme. AR-SC [99] has each sensor operate in rounds. During each round, a sensor node s_i computes its wait time, which is a representation of how much energy and contribution the sensor adds to the group. The shorter the wait time, the more contribution the sensor has. During the wait period, s_i may receive decisions from neighbors about the sensing range they will choose. The decisions are then used to update s_i's uncovered target set and wait time. Once s_i's wait time is up, s_i will select the minimum sensing range that can cover all of the uncovered targets and transmit this information to its neighbors. This approach was then extended in the *Adjustable Sensing Range Connected Sensor Cover* (ASR-CSC) algorithm [100] to allow for connectivity. Here, s_i first constructs a virtual backbone of the network and determines its responsibility. The sensor may choose to be active only for transmission purposes. Then the AS-RC heuristic is used to determine the nodes sensing range. Therefore, the sensor could be inactive, active, or active only for transmission.

The *Variable Radii Connected Sensor Cover* (VRCSC) algorithm [104] uses a Voronoi-based algorithm that partitions the region into a Voronoi Graph and selects the sensing and communication ranges of each node to ensure κ-coverage and κ-connectivity. This is a localized algorithm where each sensor makes decisions based only upon local neighborhood information. Each node computes its local and neighbors voronoi cells. Then, the node computes its sleeping benefit based on whether it decides to sleep or be active. Next, it checks whether it can be removed from the communication graph. If it can be removed and the node maximizes the sleep benefit, the node will become inactive. If not, the node will select the minimum sensing range to occupy the Voronoi cell. A similar approach was presented in the *Sensor Activation and Radius Adaptiation* (SARA) algorithm [105], however, this problem utilized Voronoi-Laguerre diagrams. Each node first constructs their Voronoi-Laguerre polygon, which represents the nodes coverage area that is not covered by its neighbors. Combining the polygon and the energy gain, the sensor node then determines if it is a redundant sensor, its sensing range is too small or its sensing range is too large. Once it determines the best sensing range to cover the polygon, it transmits its decision to its neighbors.

Two more distributed heuristics, *Adjustable Range Load Balancing Protocol* (ALBP) and *Adjustable Range Deterministic Energy Efficient Protocol* (ADEEPS), were presented by Dhawan et. al. [107] that incorporate three operating conditions for each node, active, idle, or deciding. Idle is used to conserve energy when other sensors are covering the target; Deciding is a transition phase between active and idle where the sensing range can be adjusted; and Active has the sensor use its maximum range to cover any remaining uncovered targets. The ALBP heuristic aims to balance the energy depletion while ADEEPS utilizes load balancing and reliability.

Encompassed in all of the MNLAR proposed solutions is the assumption that all of the targets in the network are fixed and their locations are known *a priori* by all of the nodes. However, in target tracking applications, the targets are dynamic and may randomly appear and disappear within the network. If the targets present were to change their locations in the network, the proposed solutions would be activating the incorrect sensor nodes and the objective of the mission would be compromised since target estimation is nonexistent. Therefore, this paper aims to solve the MNLAR problem for dynamic targets where the target locations are unknown *a priori*.

Additionally, the proposed MNLAR problems presented above do not consider sensor failures. Fault-tolerance is only proactive where the network deploys redundant sensor nodes. Thus, if a single node or multiple co-located nodes fail and create a coverage gap around the targets locations, the network will fail to meet it's objective causing a significant decrease in network lifetime.

Therefore, this chapter aims to identify coverage gaps to ensure target coverage.

Also, some of the approaches formulate the cover sets such that only $\kappa = 1$ sensors are observing each target. This formulation is appropriate when the target locations are fixed and known, however, once the targets become dynamic, less sensors covering the target results in increase estimation error. Therefore, this paper develops a distributed sensor selection method that activates $\kappa > 1$ nodes to cover the target during each time step for accurate state estimation.

5.2.3 Gaps in the literature

The following gaps have been found in the literature that are addressed in this chapter.

1. A passive coverage gap identification method that does not utilize data traffic does not exist,
2. A reactive fault recovery method for stationary DSN that is resilient to single node and multiple co-located node failures that create coverage gaps $\leq R_L$,
3. A solution to the MNLAR problem for dynamic targets with unknown location *a priori*, and
4. Using network density information to adapt the network control strategy does not exist.

5.3 Problem Formulation - New Features

The POSE.R algorithm is setup in the same manner as described in Chapter 2. However, there are a few new features that are included for the POSE.R algorithm.

First, the range of HPS devices on each multi-modal sensor node can be adjusted by controlling the amount of power supplied to the sensors [99]. Thus, each node s_i can adjust the range of it's HPS device from L levels depending on need, such that $R_{HPS}^{s_i}(k) \in \{R_1, R_2, ...R_L\}$, where $R_1 < R_2 < ... < R_L$ and R_1 is the default sensing range under normal conditions. Here, $R_1 = R_{HPS}$ as described in the previous chapters.

Second, the total energy consumed [26] by a sensor node s_i until time k is computed as

$$E^{s_i}(k) \;=\; \sum_k \sum_j e_j^{s_i} . \chi_j^{s_i}(k) \Delta T, \qquad (5.1)$$

where $e_j^{s_i}$ denotes the rate of energy consumption per unit time by a certain device $j \in$ {DPU, LPS, HPS, TX, RX, clock}; $\chi_j^{s_i}(k) \in \{0, 1\}$ indicates whether the device is ON or OFF at time k; and ΔT is the sample time interval. The energy rates $e_j^{s_i}$ are assumed constant for all devices except for the HPS devices, where $e_{HPS}^{s_i}(k)$ depends on the sensing range as follows [44]

$$e_{HPS}^{s_i}(k) = wR_{HPS}^{s_i}(k) \tag{5.2}$$

where w is the proportionality constant. Thus, the total energy consumed by the entire sensor network until time k is given as $E_{net}(k) = \sum_{i=1}^{n} E^{s_i}(k)$.

Finally, to ensure energy-efficiency, our objective is to not cover the whole area but only the areas where the target is currently present and predicted to be during the next time step, thus enabling opportunistic sensing and coverage. However, the network may develop coverage gaps over time due to non-uniform node distribution, sensor failures, node drifting, etc., which can affect the tracking performance. Therefore, the network must cover the coverage gap, $G(k)$, defined in Chapter 2.

To achieve resilience, that is to track the target even when it passes through a coverage gap, we perform distributed optimization to extend the sensing ranges of selected *HPS* state sensors around the coverage gap where the target is approaching.

Remark 5.3.1 *By optimally extending the ranges of HPS sensors around the coverage gap, the coverage gap can reduce or disappear at the time when target is passing by.*

Definition 5.3.1 (Target Coverage) *Target coverage is said to be achieved at time k if $\mathbf{u}^{\tau_\ell}(k) \notin G(k), \forall \tau_\ell \in \mathcal{T}$, i.e. all targets are covered.*

Definition 5.3.2 (Coverage Degree) *Let $\mathcal{S}_c(\mathbf{u}^{\tau_\ell}(k)) = \{s_i \in \mathcal{S} : \mathbf{u}^{\tau_\ell}(k) \in \Omega^{s_i}(k)\}$ denote the set of all nodes in whose coverage areas the target falls in at time k. Then, the coverage degree of target τ_ℓ is defined as $D(\mathbf{u}^{\tau_\ell}(k)) = |\mathcal{S}_c(\mathbf{u}^{\tau_\ell}(k))|$.*

Definition 5.3.3 (Base Coverage Degree) *Let $S_{c,R_1}(\mathbf{u}^{\tau_\ell}(k)) = \{s_i \in S_c(\mathbf{u}^{\tau_\ell}(k)) : R_{HPS}^{s_i} = R_1\}$ be the set of nodes covering τ_ℓ with their base sensing range R_1. Then, the base coverage degree of a target τ_ℓ is defined as $D_{R_1}(\mathbf{u}^{\tau_l}(k)) = |S_{c,R_1}(\mathbf{u}^{\tau_\ell}(k))|$.*

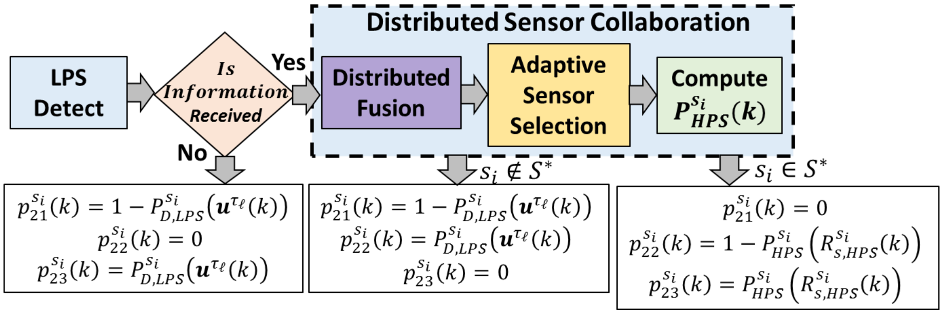

Distributed Sensor Collaboration
LPS Detect
Is Information Received
Yes
No
Distributed Fusion
Adaptive Sensor Selection
Compute $P_{HPS}^{s_i}(k)$
$s_i \notin S^*$
$s_i \in S^*$
$p_{21}^{s_i}(k) = 1 - P_{D,LPS}^{s_i}(\boldsymbol{u}^{\tau_\ell}(k))$
$p_{22}^{s_i}(k) = 0$
$p_{23}^{s_i}(k) = P_{D,LPS}^{s_i}(\boldsymbol{u}^{\tau_\ell}(k))$
$p_{21}^{s_i}(k) = 1 - P_{D,LPS}^{s_i}(\boldsymbol{u}^{\tau_\ell}(k))$
$p_{22}^{s_i}(k) = P_{D,LPS}^{s_i}(\boldsymbol{u}^{\tau_\ell}(k))$
$p_{23}^{s_i}(k) = 0$
$p_{21}^{s_i}(k) = 0$
$p_{22}^{s_i}(k) = 1 - P_{HPS}^{s_i}\left(R_{s,HPS}^{s_i}(k)\right)$
$p_{23}^{s_i}(k) = P_{HPS}^{s_i}\left(R_{s,HPS}^{s_i}(k)\right)$

Figure 5.2: Low Power Sensing State Algorithm

Therefore, the objective of this chapter is to develop a network autonomy approach that employs a distributed node-level supervisor that probabilistically controls the multi-modal sensor node's devices to meet the following requirements: (1) improved network life expectancy via opportunistic sensing, (2) resilient coverage of the targets in the presence of coverage gaps, and (3) high tracking accuracy and low missed detection rates.

5.4 POSE.R Algorithms

Consider a sensor node $s_i \in \mathcal{S}$ that has a PFSA-based supervisor as described in Chapter 3.3, except that the algorithms within the states now perform the following algorithms.

5.4.1 Sleep State Description

The *Sleep* state, θ_1, is used to minimize energy consumption by disabling all devices on s_i except for a clock to allow for state transitions. A node s_i probabilistically switches to this state if a target is located far away from this node or if this node is not selected for tracking. After every time interval ΔT, s_i may continue in the *Sleep* state with a probability $p_{11}^{s_i} = p_{sleep}$ or transition to the *LPS* state with $p_{12}^{s_i} = 1 - p_{sleep}$, where $p_{sleep} \in [0, 1]$ is a design parameter. From the *Sleep* state, node s_i cannot go the *HPS* state directly, i.e. $p_{13}^{s_i} = 0$ because it does not have any information about the targets location.

5.4.2 Low Power Sensing (LPS) State Description

The *LPS* state, θ_2, is designed to allow for target detection while conserving energy. In this state, the DPU, TX, RX, and the LPS devices are enabled. The flowchart for the algorithm is shown in Fig. 5.2 and described below.

• ***Target Detection***: Target detection is achieved by either (i) using the LPS devices or (ii) fusing the information received from neighbors. If a target is located within R_{LPS}, s_i detects it with a probability $P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$ according to Eq. (2.5).

Next, s_i checks if it has received any information from *HPS* state sensors in its neighborhood. Let $\mathcal{N}^{s_i}_{HPS} \subseteq \mathcal{N}^{s_i}$ be the set of nodes in the *HPS* state that have broadcasted target state information to s_i. If no information is received, i.e. $\mathcal{N}^{s_i}_{HPS} = \emptyset$, then s_i will transition to the *HPS* state solely based on its own $P^{s_i}_{D,LPS}(\mathbf{u}^{\tau_\ell}(k))$. If information is received, i.e. $\mathcal{N}^{s_i}_{HPS} \neq \emptyset$, then s_i performs distributed sensor collaboration to make an informed switching decision as summarized below.

• ***Distributed Sensor Collaboration***: This consists of three steps (full details are in Section 5.5) as follows:

1. *Distributed Fusion*: This step fuses the received information and obtain a target state estimates and predictions.

2. *Adaptive Sensor Selection*: The predicted state estimate, $\hat{\mathbf{x}}^{s_i}(k+1|k)$, is used to i) select the optimal set of sensors, $\mathcal{S}^*$, to track the target during the next time step, and (ii) select their optimal sensing ranges, $\mathcal{R}^* = \{R^{s_j}_{HPS}(k+1)^*, \forall s_j \in \mathcal{S}^*\}$, to ensure target coverage. Additionally, the base coverage degree, $D_{R_1}(H\hat{\mathbf{x}}^{s_i}(k+1|k))$, is computed to identify if the selected sensors must expand their sensing ranges to achieve N_{sel}-coverage.

3. *Computation of State Transition Probability to* HPS *state*: If $s_i \in \mathcal{S}^*$, the node computes a probability $P^{s_i}_{HPS}(k)$ (see Eq. 3.12) to transition to the *HPS* state. However, if $s_i \notin \mathcal{S}^*$, this implies that there are other better sensors to track the target. In this case, s_i will probabilistically stay in *LPS* state to stay aware of the target and participate in sensor selection during the next time step if applicable. When the node is located within R_1 of the target's predicted

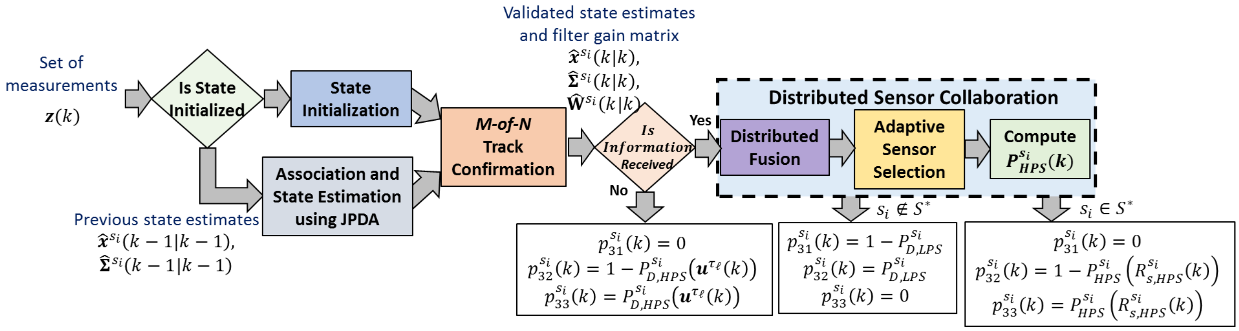

Set of measurements
$\mathbf{z}(k)$
Is State Initialized
State Initialization
Association and State Estimation using JPDA
Previous state estimates
$\hat{\mathbf{x}}^{s_i}(k-1|k-1)$,
$\hat{\mathbf{\Sigma}}^{s_i}(k-1|k-1)$
M-of-N Track Confirmation
Validated state estimates and filter gain matrix
$\hat{\mathbf{x}}^{s_i}(k|k)$,
$\hat{\mathbf{\Sigma}}^{s_i}(k|k)$,
$\hat{\mathbf{W}}^{s_i}(k|k)$
Is Information Received
Yes
No
Distributed Sensor Collaboration
Distributed Fusion
Adaptive Sensor Selection
Compute $P_{HPS}^{s_i}(k)$
$s_i \notin S^*$
$s_i \in S^*$
$p_{31}^{s_i}(k) = 0$
$p_{32}^{s_i}(k) = 1 - P_{D,HPS}^{s_i}(\mathbf{u}^{\tau_\ell}(k))$
$p_{33}^{s_i}(k) = P_{D,HPS}^{s_i}(\mathbf{u}^{\tau_\ell}(k))$
$p_{31}^{s_i}(k) = 1 - P_{D,LPS}^{s_i}$
$p_{32}^{s_i}(k) = P_{D,LPS}^{s_i}$
$p_{33}^{s_i}(k) = 0$
$p_{31}^{s_i}(k) = 0$
$p_{32}^{s_i}(k) = 1 - P_{HPS}^{s_i}\left(R_{s,HPS}^{s_i}(k)\right)$
$p_{33}^{s_i}(k) = P_{HPS}^{s_i}\left(R_{s,HPS}^{s_i}(k)\right)$

Figure 5.3: High Power Sensing State Algorithm

location, s_i should stay aware in the *LPS* state. However, if the node is located a distance $> R_1$ from the target, s_i should only stay aware if the base coverage degree is insufficient, i.e. $D_{R_1}(H\hat{\mathbf{x}}^{s_i}(k+1|k)) < N_{sel}$. This indicates that the selected nodes within R_L of the target must expand their sensing ranges to achieve $D(H\hat{\mathbf{x}}^{s_i}(k+1|k)) = N_{sel}$. Therefore, the nodes not selected within R_L should maintain awareness in the *LPS* state in case they are a future candidate to track the target.

The probabilistic control of the *LPS* state, θ_2, is described in Alg. 1.

5.4.3 High Power Sensing (HPS) State Description

The *HPS* state, θ_3, is designed to cover and estimate the target's state using measurements from its HPS devices. In θ_3, s_i enables the HPS devices, DPU, TX, and RX and performs the algorithm shown in Fig. 5.3.

• ***Data Association and State Estimation***: In the *HPS* state, node s_i first receives a set of measurements, $\mathbf{z}(k)$, from its HPS devices of a target or clutter within a range $R_{HPS}^{s_i}(k)$, where $R_{HPS}^{s_i}(k)$ was selected during the previous time step in the *LPS* or *HPS* state. Subsequently, the previous state and covariance estimates, $\hat{\mathbf{x}}^{s_i}(k-1|k-1)$, $\hat{\mathbf{\Sigma}}^{s_i}(k-1|k-1)$, are updated using the *Joint Probabilistic Data Association* (JPDA) method [48] to generate $\hat{\mathbf{x}}^{s_i}(k|k)$ and $\hat{\mathbf{\Sigma}}^{s_i}(k|k)$. If the measurements do not associate to a previous state estimate, s_i initializes a new state estimate [71]. Details are presented in Chapter 3.3.2.

• ***M-of-N Track Confirmation***: The measurements of the HPS devices may contain false alarms due to clutter. This causes s_i to initialize a new state estimate when clutter does not associate to

```
input  : $\mathbf{u}^{s_i}$, $\mathcal{N}^{s_i}_{HPS}$, $\hat{\mathbf{I}}^{s_i}(k)$, $N_{sel}$, $\theta^{s_i}(k)$, and $H$
output: $p^{s_i}_{idx,1}(k), p^{s_i}_{idx,2}(k)$, and $p^{s_i}_{idx,3}(k)$
if $\theta^{s_i}(k) = \theta_2$ then                                      // $s_i$ is in the LPS state
    $idx \leftarrow 2$
else if $\theta^{s_i}(k) = \theta_3$ then                                 // $s_i$ is in the HPS state
    $idx \leftarrow 3$
end
if $\mathcal{N}^{s_i}_{HPS} = \emptyset$ then                             // Information was not received by $s_i$
    if $idx = 2$ then                                                     // $s_i$ is in the LPS state
        $p^{s_i}_{idx,1}(k) \leftarrow 1 - P^{s_i}_{D,LPS}(H\hat{\mathbf{x}}^{s_i}(k+1|k))$,
        $p^{s_i}_{idx,2}(k) \leftarrow 0$,
        $p^{s_i}_{idx,3}(k) \leftarrow P^{s_i}_{D,LPS}(H\hat{\mathbf{x}}^{s_i}(k+1|k))$
    else if $idx = 3$ then                                                // $s_i$ is in the HPS state
        $p^{s_i}_{idx,1}(k) \leftarrow 0$,
        $p^{s_i}_{idx,2}(k) \leftarrow 1 - P^{s_i}_{D,HPS}(R^{s_i}_{HPS}(k))$,
        $p^{s_i}_{idx,3}(k) \leftarrow P^{s_i}_{D,HPS}(R^{s_i}_{HPS}(k))$
    end
else if $\mathcal{N}^{s_i}_{HPS} \neq \emptyset$ then                     // Information was received by $s_i$
    // Call distributed sensor collaboration algorithm
    $[\hat{\mathbf{x}}^{s_i}(k+1|k), \mathcal{S}^*, \mathcal{R}^*] \leftarrow DSC(\hat{\mathbf{I}}^{s_i}(k))$
    Compute $D_{R_1}(H\hat{\mathbf{x}}^{s_i}(k+1|k))$ using $\mathcal{S}^*$ and $\mathcal{R}^*$
    if $s_i \in \mathcal{S}^*$ then                                       // Node $s_i$ was selected
        $p^{s_i}_{idx,1}(k) \leftarrow 0$,
        $p^{s_i}_{idx,2}(k) \leftarrow 1 - P^{s_i}_{HPS}(k, R^{s_i}_{HPS}(k+1))$,
        $p^{s_i}_{idx,3}(k) \leftarrow P^{s_i}_{HPS}(k, R^{s_i}_{HPS}(k+1))$
    else if $s_i \notin \mathcal{S}^*$ then                               // Node $s_i$ was not selected
        if $||\mathbf{u}^{s_i} - H\hat{\mathbf{x}}^{s_i}(k+1|k)|| \leq R_1$ then    // The target's predicted position is within $R_1$ of $s_i$
            $p^{s_i}_{idx,1}(k) \leftarrow 1 - P^{s_i}_{D,LPS}(H\hat{\mathbf{x}}^{s_i}(k+1|k))$,
            $p^{s_i}_{idx,2}(k) \leftarrow P^{s_i}_{D,LPS}(H\hat{\mathbf{x}}^{s_i}(k+1|k))$,
            $p^{s_i}_{idx,3}(k) \leftarrow 0$
        else if $||\mathbf{u}^{s_i} - H\hat{\mathbf{x}}^{s_i}(k+1|k)|| > R_1$ then  // The target's predicted position is further than $R_1$ of $s_i$
            if $D_{R_1}(H\hat{\mathbf{x}}^{s_i}(k+1|k)) = N_{sel}$ then   // Base coverage degree is sufficient
                $p^{s_i}_{idx,1}(k) \leftarrow 1$,
                $p^{s_i}_{idx,2}(k) \leftarrow 0$,
                $p^{s_i}_{idx,3}(k) \leftarrow 0$
            else if $D_{R_1}(H\hat{\mathbf{x}}^{s_i}(k+1|k)) < N_{sel}$ then  // Base coverage degree is insufficient
                $p^{s_i}_{idx,1}(k) \leftarrow 1 - P^{s_i}_{D,HPS}(R^{s_i}_{HPS}(k))$,
                $p^{s_i}_{idx,2}(k) \leftarrow P^{s_i}_{D,HPS}(R^{s_i}_{HPS}(k))$,
                $p^{s_i}_{idx,3}(k) \leftarrow 0$
            end
        end
    end
end
```

Algorithm 1: Probabilistic Control of *LPS* and *HPS* states

a previous estimate. To account for false alarms and ensure that a false track is not propagated throughout the network, s_i utilizes the $M - of - N$ *Track Confirmation Logic* [72] (presented in Chapter 3.3.2) to allow the network to be robust to false alarms.

Subsequently, the confirmed target track's state and covariance estimates, $\hat{\mathbf{x}}^{s_i}(k|k)$ and $\hat{\mathbf{\Sigma}}^{s_i}(k|k)$, and filter gain matrix, $\hat{\mathbf{W}}^{s_i}(k)$, are broadcasted to its neighbors. Next, s_i checks if it has received any information from *HPS* state sensors $\mathcal{N}_{HPS}^{s_i}$ in its neighborhood. Since s_i is in the *HPS* state and has broadcasted information to its neighbors, the set of *HPS* state sensors becomes $\mathcal{N}_{HPS}^{s_i} = \mathcal{N}_{HPS}^{s_i} \cup \{s_i\}$. However, if s_i has not transmitted a confirmed track, $\mathcal{N}_{HPS}^{s_i}$ does not include s_i. If no information is received, i.e. $\mathcal{N}_{HPS}^{s_i} = \emptyset$, then s_i will rely on it's own probability of detection, γ_{HPS}, to remain in the *HPS* state. If information is received, i.e. $\mathcal{N}_{HPS}^{s_i} \neq \emptyset$, then s_i performs the distributed sensor collaboration to make an informed switching decision.

• ***Distributed Sensor Collaboration***: Collaboration is achieved using the same process as Section 5.4.2. This includes distributed fusion, adaptive sensor selection, and computation of state transition probabilities.

The probabilistic control at the *HPS* state, θ_3, is described in Alg. 1.

5.5 Distributed Sensor Collaboration

The distributed sensor collaboration algorithm, consists of distributed fusion, adaptive sensor selection, and computation of the probability of detecting the target's predicted location using the HPS devices as follows:

5.5.1 Distributed Fusion

The first step in sensor collaboration consists of fusing the transmitted target state information together to allow node s_i an understanding of the target's location. This allows for (1) improved state estimation and (2) target detection without sensing (if node s_i is in the *LPS* state). Consider a node s_i which could be in the *LPS* or *HPS* state. The information ensembles it received consists of

$$\hat{\mathbf{I}}^{s_i}(k) = \{(\hat{\mathbf{x}}^{s_j}(k|k), \hat{\boldsymbol{\Sigma}}^{s_j}(k|k), \hat{\mathbf{W}}^{s_j}(k)), \forall s_j \in \mathcal{N}_{HPS}^{s_i}\}, \tag{5.3}$$

where $\hat{\mathbf{x}}^{s_j}(k|k)$, $\hat{\boldsymbol{\Sigma}}^{s_j}(k|k)$, and $\hat{\mathbf{W}}^{s_j}(k)$ correspond to the target state, covariance, filter gain estimates of s_j at time k.

• ***Trustworthy Set Formation*** Due to noise and other factors, the information received must first be validated to ensure that it is accurate and reliable before processing. Since false tracks produced by clutter typically deviate from the target motion model, the covariance of the estimate may be very high. Therefore, this step aims to further reduce false tracks by forming a set of trustworthy neighbors $\mathcal{N}_T^{s_i} \subseteq \mathcal{N}_{HPS}^{s_i}$. This is achieved by evaluating the sum of the position error as follows

$$\mathcal{N}_T^{s_i} = \{s_j \in \mathcal{N}_{HPS}^{s_i} : Trace(\hat{\mathbf{\Sigma}}_z^{s_i,c}(k+1|k)) \leq \xi\} \tag{5.4}$$

where $\hat{\mathbf{\Sigma}}_z^{s_i,c}(k+1|k) = \mathbf{H}(k)\hat{\mathbf{\Sigma}}^{s_j}(k|k)\mathbf{H}(k)'$ is the target's predicted location; $\mathbf{H}(k)$ is the Jacobian of the measurement model defined in Eq. (2.6); and ξ is the maximum tolerance of the estimate. In this paper, $\xi = \frac{R_1^2\sigma_\phi^2+\sigma_R^2}{2}$, where σ_ϕ and σ_R are the standard deviations in the azimuth and range measurements of the HPS sensor. Finally, node s_i receives the following trustworthy information:

$$\hat{\mathbf{I}}_T^{s_i}(k) = \left\{\left(\hat{\mathbf{x}}^{s_j}(k|k), \hat{\mathbf{\Sigma}}^{s_j}(k|k), \hat{\mathbf{W}}^{s_j}(k)\right), \forall s_j \in \mathcal{N}_T^{s_i}\right\}, \tag{5.5}$$

• ***Track-to-Track Association and Fusion*** Next, the trustworthy information is associated to ensure that it is related to the same target to further improve fusion. In this work, the *Track-to-Track Association Method* [69] is used for this purpose. Details are presented in Chapter 3.3.2. This method associates the trustworthy information into C different groups which correspond to the C different targets that could be present within the node s_i's neighborhood; thus forming the information ensembles: $\hat{\mathbf{I}}_T^{s_i,c}(k) \subseteq \hat{\mathbf{I}}_T^{s_i}(k)$, where $c = 1, ..., C$. Subsequently, for each c, the state information in $\hat{\mathbf{I}}_T^{s_i,c}(k)$ is fused to form a single state ($\hat{\mathbf{x}}^{s_i,c}(k|k)$) and covariance ($\hat{\mathbf{\Sigma}}^{s_i,c}(k|k)$) estimate, using the *Track-to-Track Fusion* (T2TF) algorithm [70]. Details are presented in Chapter 3.3.2.

• ***Target Prediction*** Once all the received state information is fused, node s_i computes a one-step prediction [45] of the target's state using the Extended Kalman Filter:

$$\begin{aligned}
\hat{\mathbf{x}}^{s_i,c}(k+1|k) &= \mathbf{f}(\hat{\mathbf{x}}^{s_i,c}(k|k), k) \\
\hat{\mathbf{\Sigma}}^{s_i,c}(k+1|k) &= \mathbf{F}(k)\hat{\mathbf{\Sigma}}^{s_i,c}(k|k)\mathbf{F}(k)' + \mathbf{Q},
\end{aligned} \tag{5.6}$$

where $\mathbf{F}(k)$ is the Jacobian of the state transition matrix evaluated at $\hat{\mathbf{x}}^{s_i,c}(k|k)$ and $\mathbf{Q}$ is the process noise covariance error.

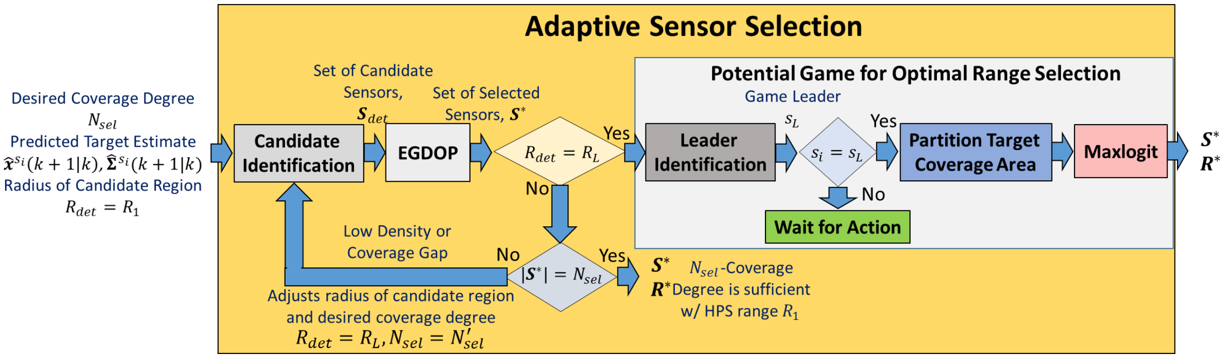


Figure 5.4: Adaptive Sensor Selection Algorithm

5.5.2 Adaptive Sensor Selection

Next, the predicted state of each target is used to select N_{sel} nodes and their sensing ranges to track the target in the *HPS* state during the next time step to ensure target coverage. Let $\mathcal{N}_{dsc}^{s_i} \subseteq \mathcal{N}^{s_i} \cup \{s_i\}$ be the set of nodes that are in either the *LPS* or *HPS* state who have received target information and are performing sensor collaboration. Each $s_j \in \mathcal{N}_{dsc}^{s_i}$ will perform the adaptive sensor selection algorithm, shown in Fig. 5.4, which is a multi-stage filter process that identifies the optimal sensors and their sensing ranges for resilient coverage. Therefore, the following steps are computed by node s_i for each target τ_c.

• ***Candidate Identification***: First, node s_i uses the target's predicted state and covariance estimates, $\hat{\mathbf{x}}^{s_i,c}(k+1|k)$ and $\hat{\mathbf{\Sigma}}^{s_i,c}(k+1|k)$, to identify the nodes that can completely cover the target's predicted location. Let $\Omega_{can} \in \Omega$ be the region surrounding the target's predicted location s.t. if any node lies within Ω_{can}, it can cover 6σ of the predicted target distribution with a sensing range R_{det}. More formally,

$$\Omega_{can} = \quad \{(x,y);(\tfrac{x-\hat{x}^{s_i,c}(k+1|k)}{R_{det}-3\sigma_x})^2 + (\tfrac{y-\hat{y}^{s_i,c}(k+1|k)}{R_{det}-3\sigma_y})^2 \leq 1\} \tag{5.7}$$

where σ_x and σ_y are the x and y position standard deviations of the uncertainty estimate $\hat{\mathbf{\Sigma}}_z^{s_i,c}(k+1|k)$, while $\hat{x}^{s_i,c}(k+1|k)$ and $\hat{y}^{s_i,c}(k+1|k)$ are the x and y predicted position estimates of $\hat{\mathbf{x}}^{s_i,c}(k+1|k)$.

Initially, $R_{det} = R_1$ is used to identify if there are a sufficient number of nodes available to cover the target with the minimum HPS sensing range to minimize energy consumption. Therefore, the set of sensors that can cover the target's predicted location distribution are defined as:

$$\mathcal{S}_{can} = \{s_j \in (\mathcal{N}^{s_i} \cup s_i); \mathbf{u}^{s_j} \in \Omega_{can}\}. \tag{5.8}$$

Next, let $\mathcal{S}_{det} = \{\mathcal{N}_{dsc}^{s_i} \cap \mathcal{S}_{can}\}$ be the set of nodes that are in the *LPS* or *HPS* state. If node $s_i \in \mathcal{S}_{det}$, then it will broadcast it's percent energy remaining, $E_R^{s_i}(k) = \frac{E_0 - E^{s_i}(k)}{E_0}$, to it's neighbors to indicate that it is available for tracking, where E_0 is the node's initial energy and $E^{s_i}(k)$ is given in Eq. (2.3). Additionally, the remaining nodes in $\mathcal{S}_{det}$ broadcast their energy remaining, and node s_i receives their energy information and forms the set $\mathbf{E}_{det} = \{E_R^{s_j}(k), \forall s_j \in \mathcal{S}_{det}\}$. The set $\mathcal{S}_{det}$ is identified dynamically by node s_i based on it's neighbors that have transmitted their energy information.

• ***Energy-based Geometric Dilution of Precision***: Next, node s_i computes the best set of sensors around the target's predicted position that maximizes the energy remaining while minimizing the mean squared error of the target estimate. However, maximizing the energy remaining typically does not guarantee that the mean squared error will be minimized (and vice versa). Therefore, to jointly optimize these two criteria, the *Energy-based Geometric Dilution of Precision* (EGDOP) measure is proposed.

First, s_i identifies the node s_j s.t. $s_j = \arg\max_{s_h \in \mathcal{S}_{det}}(E_R^{s_h}(k))$ and creates the set $\mathcal{S}^* = \{s_j\}$. Then, node s_i incrementally adds a node $s_h \notin \mathcal{S}^*$, one at a time, to the set $\mathcal{S}^*$ s.t. the following utility function is maximized and $|\mathcal{S}^*| = \min(N_{sel}, |\mathcal{S}_{det}|)$.

$$\mu(\mathcal{S}^*) = \frac{det(\mathbf{J}(\mathcal{S}^*))}{trace(\mathbf{J}(\mathcal{S}^*))}, \tag{5.9}$$

$$\mathbf{J}(\mathcal{S}^*) = \sum_{s_h \in \mathcal{S}^*} \frac{E_R^{s_h}(k)}{\sigma_{\theta,n}^2 r_{s_h,n}^2} \begin{bmatrix} s^2\phi_{s_h} & -s\phi_{s_h} c\phi_{s_h} \\ -s\phi_{s_h} c\phi_{s_h} & c^2\phi_{s_h} \end{bmatrix}, \tag{5.10}$$

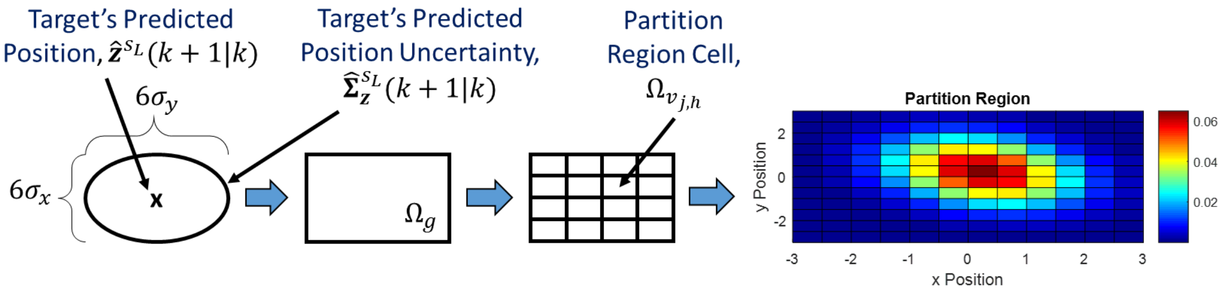
where $s \equiv sin$ and $c \equiv cos$; $r_{s_j,n} = (\frac{x - \hat{x}^{s_i,c}(k+1|k)}{R_{det} - 3\sigma_x})^2 + (\frac{y - \hat{y}^{s_i,c}(k+1|k)}{R_{det} - 3\sigma_y})^2$ is the normalized range of sensor s_j to the target's predicted position; $\sigma_{\theta,n} = \frac{\sigma_\theta}{2\pi}$ is the normalized measurement angle standard deviation; and ϕ_{s_j} is the azimuth angle between sensor s_j and the target's predicted position.

• ***Coverage Degree Identification***: Next, node s_i determines the coverage degree of the target's predicted position and utilizes this information to adapt the sensor selection algorithm. First, s_i determines the coverage degree by computing the length of the set $\mathcal{S}^*$. If $|\mathcal{S}^*| = N_{sel}$, the adaptive sensor selection algorithm is complete since the base coverage degree, $D_{R_1}(\mathbf{u}^{\tau_\ell}(k))$, is satisfied. Therefore, the sets $\mathcal{S}^*$ and $\mathcal{R}^*$ are outputted where each sensor in $\mathcal{S}^*$ uses the HPS range R_1. However, if $|\mathcal{S}^*| < N_{sel}$, the region Ω_{can} must be expanded to identify nodes that can extend their sensing range to ensure that the coverage degree is $D(\mathbf{u}^{\tau_\ell}(k)) = N_{sel}$.

Consequently, node s_i restarts the adaptive sensor selection algorithm and expands the candidate region, Ω_{can}, to include nodes that can detect the target with the maximum sensing range R_L. This results in a larger candidate set that consists of nodes that can detect the target with sensing ranges between $[R_1, R_L]$. Therefore, the new sensor selection approach increases the complexity by jointly optimizing the sensing ranges, number of sensors selected, geometric diversity, and network energy consumption. However, solving the joint optimization problem is very complex and may not be achieved in real time. To simplify this, a multi-stage filtering process is employed, that first selects $N'_{sel} > N_{sel}$ sensors that have the most energy remaining and are geometrically diverse using EGDOP; then, s_i selects the optimal sensing ranges for the N'_{sel} nodes using potential games s.t. the coverage degree of the target is $D(\mathbf{u}^{\tau_\ell}(k)) = N_{sel}$.

Therefore, s_i restarts the Candidate Identification and EGDOP processes with the following new inputs: $\hat{\mathbf{x}}^{s_i}(k+1|k)$; $\hat{\mathbf{\Sigma}}^{s_i}(k+1|k)$; $R_{det} = R_L$; and $N_{sel} = N'_{sel}$ where $N'_{sel} > N_{sel}$ is a design parameter. This then produces a new set of candidates, $\mathcal{S}_{det}$, and optimal sensors, $\mathcal{S}^*$, that are geometrically diverse, have sufficient energy, and can cover the target's predicted location. Then, if node $s_i \notin \mathcal{S}^*$, s_i quits the sensor selection process and continues to update its state transition probabilities. However, if node $s_i \in \mathcal{S}^*$, the node moves forward to select it's optimal sensing range.

• ***Leader Identification*** The first step in range selection is to identify the best node in $\mathcal{S}^*$ to compute the optimal sensing ranges for the group. A leader is selected because a game requires frequent communication between the players which costs high energy. Therefore, one node is selected to perform sensor range selection to reduce energy consumption. Leader selection is per-



Target's Predicted Position, $\hat{\mathbf{z}}^{s_L}(k+1|k)$
Target's Predicted Position Uncertainty, $\hat{\mathbf{\Sigma}}_{\mathbf{z}}^{s_L}(k+1|k)$
Partition Region Cell, $\Omega_{v_{j,h}}$
$6\sigma_y$
$6\sigma_x$
x
Ω_g
Partition Region
y Position
x Position
2
0
-2
-3
-2
-1
0
1
2
3
0.06
0.04
0.02

Figure 5.5: Target Coverage Partition Region

formed independently on each node in $\mathcal{S}^*$ and is found based on the maximum energy remaining. Therefore, node s_i identifies the leader s_L as such:

$$s_L = \arg\max_{s_j \in \mathcal{S}^*}(E_R^{s_j}(k)). \tag{5.11}$$

If $s_i = s_L$, node s_i is the range selection leader and will continue to the next step, while if $s_i \neq s_L$, s_i waits until node s_L computes the optimal ranges and transmits the result.

• ***Partition Target Coverage Area***: To identify the optimal ranges, the leader s_L first identifies the target coverage area based on the target's predicted state and covariance estimates, $\hat{\mathbf{x}}^{s_L,c}(k+1|k)$ and $\hat{\mathbf{\Sigma}}^{s_L,c}(k+1|k)$. As shown in Fig. 5.5, let $\Omega_g \in \Omega$ be the target coverage area which is modeled as the rectangular area that contains 6σ of the target's predicted location defined as:

$$\begin{aligned}\Omega_g \quad = \quad & \{(x,y) \in \Omega; -3\sigma_x \leq ||x - \hat{x}^{s_L,c}(k+1|k)|| \leq 3\sigma_x, \\ & -3\sigma_y \leq ||y - \hat{y}^{s_L,c}(k+1|k)|| \leq 3\sigma_y\}.\end{aligned} \tag{5.12}$$

Therefore, optimizing over the target coverage area ensures that the target will be covered.

Then, Ω_g is partitioned into $V \cdot V$ cells, where each cell is defined as $\Omega_{v_{j,h}}, \forall j = h = 1, ..., V$. Next, each $\Omega_{v_{j,h}}$ is assigned a worth based on the multivariate normal probability density function.

$$v_{j,h} = \mathcal{N}([x_{j,h}^c, y_{j,h}^c]', \hat{\mathbf{z}}^{s_L,c}(k+1|k), \hat{\mathbf{\Sigma}}_z^{s_L,c}(k+1|k))/c_1 \tag{5.13}$$

where $[x^c_{j,h}, y^c_{j,h}]$ is the center point of the cell $\Omega_{v_{j,h}}$; and

$$c_1 = \sum_{j=1}^{V} \sum_{h=1}^{V} \mathcal{N}([x^c_{j,h}, y^c_{j,h}]', \hat{\mathbf{z}}^{s_L,c}(k+1|k), \hat{\mathbf{\Sigma}}_z^{s_L,c}(k+1|k))$$

is a normalizing constant s.t. $\sum_{j=1}^{V} \sum_{h=1}^{V} v_{j,h} = 1$. The worth represents the probability of the target being located in the cell $\Omega_{v_{j,h}}$, as shown in Figure 5.5 on the right. The partition region, Ω_g, cell location, $[x^c_{j,h}, y^c_{j,h}]'$, and worth, $v_{j,h}$, are then used to identify the optimal sensing ranges to ensure that the target coverage area is covered.

• ***Optimal Range Selection using Potential Games***: Next, the leader s_L computes the optimal sensing range for each node $s_j \in \mathcal{S}^\star$, such that N_{sel} sensors are covering each cell of the partition region Ω_g while minimizing s_j's sensing range, i.e. redundant coverage. However, it is difficult to optimize over all combination of joint ranges in real time. Therefore, to achieve range selection, this work implements a game-theoretic approach using the concept of *Potential Games*.

The use of potential games provides the following advantages: (i) at least one pure Nash Equilibrium is guaranteed to exist; (ii) there exist learning algorithms that can asymptotically converge to the optimal equilibrium with a fast convergence rate, e.g., the Max-Logit algorithm; and (iii) the utility of each player is perfectly aligned with a global objective function, namely the *potential function*, such that when the players negotiate to maximize their own utilities, the global potential function is also maximized.

Preliminaries: A game G in strategic form [108] consists of:

- A finite set of players. In this work, the set of players are the sensor nodes selected using EGDOP, $\mathcal{S}^\star$.

- A non-empty set of actions $\mathcal{A}_i$ associated to each player $s_i \in \mathcal{S}^\star$. In this paper, each action $a_i \in \mathcal{A}_i$ indicates a different sensing range. Specifically, the action set $\mathcal{A}_i = \{R_0, R_1, ..., R_L\}$, where R_0 represents a sensing range $R^{s_j}_{HPS}(k+1) = 0m$ implying a state transition to *LPS* or *Sleep* state. The action set is assumed to be identical for all players, i.e., $\mathcal{A}_i = \mathcal{A}_j$, $\forall s_i, s_j \in \mathcal{S}^\star$.

- The *utility function* associated with each player s_i, defined as $\mathcal{U}_i : \mathcal{A}_{\mathcal{S}^\star} \to \mathbb{R}$, where $\mathcal{A}_{\mathcal{S}^\star} = \mathcal{A}_1 \times \ldots \times \mathcal{A}_{|\mathcal{S}^\star|}$ denotes the set of joint actions for all players.

The utility function computes the payoff that a sensor node $s_i \in \mathcal{S}^\star$ can expect by choosing an action $a_i \in \mathcal{A}_i$, given that the rest of the players jointly select $a_{-i} \in \mathcal{A}_{-i}$, where $\mathcal{A}_{-i} := \mathcal{A}_1 \times \ldots \times \mathcal{A}_{i-1} \times \mathcal{A}_{i+1} \times \ldots \times \mathcal{A}_{|\mathcal{S}^\star|}$. A joint action of all players $a \in \mathcal{A}$ is often written as $a = (a_i, a_{-i})$.

Definition 5.5.1 (Nash Equlibrium) *A joint action $a = (a_i^\star, a_{-i}^\star) \in \mathcal{A}$ is called a pure Nash Equilibrium if*

$$\mathcal{U}_i(a_i^\star, a_{-i}^\star) = \max_{a_i \in \mathcal{A}_i} \mathcal{U}_i(a_i, a_{-i}^\star),\ \forall s_i \in \mathcal{S}^\star \tag{5.14}$$

Definition 5.5.2 (Potential Games) *A game G in strategic form with action sets $\{\mathcal{A}_i\}_{i=1}^{|\mathcal{S}^\star|}$ together with utility functions $\{\mathcal{U}_i\}_{i=1}^{|\mathcal{S}^\star|}$ is a potential game if and only if, a potential function $\Phi : \mathcal{A} \to \mathbb{R}$ exists, s.t. $\forall\ s_i \in \mathcal{S}^\star$*

$$\mathcal{U}_i(a_i', a_{-i}) - \mathcal{U}_i(a_i'', a_{-i}) = \Phi(a_i', a_{-i}) - \Phi(a_i'', a_{-i}) \tag{5.15}$$

$\forall\ a_i', a_i'' \in \mathcal{A}_i$ and $\forall\ a_{-i} \in \mathcal{A}_{-i}$.

A potential game requires the perfect alignment between the utility of each player and the globally shared potential function Φ for all players, i.e., the change in $\mathcal{U}_i$ by unilaterally deviating the action of player s_i is equal to the amount of change in the potential function Φ. In this regard, as the players negotiate towards maximizing their individual utilities, the global objective is also optimized.

The Sensor Range Selection Game: In order to find the joint action that maximizes coverage of the partition region, Ω_g, while minimizing redundant coverage, the potential function is designed as follows.

$$\Phi(a) = \sum_{j=1}^{V}\sum_{h=1}^{V} v_{j,h} B_{j,h}(a) - \frac{\sum_{s_j \in \mathcal{S}^*} E_c(a_j)}{|\mathcal{S}^*| E_c(R_L)} \tag{5.16}$$

where $v_{i,j}$ is given in Eq. (5.13), $B_{j,h}(a)$ is the coverage function, and the energy cost by taking the action a_j is

$$E_c(a_j) = \begin{cases} \Delta T \cdot e^{s_j}_{HPS}(a_j) & \text{if } a_j \neq R_0 \\ \Delta T \cdot e_{LPS} & \text{if } a_j = R_0 \end{cases} \tag{5.17}$$

The first term on the right hand side of Eq. 5.16 computes the target coverage, while the second term evaluates redundant coverage achieved with the joint action $a \in \mathcal{A}$. Depending on the requirements of the DSN, the design of the coverage function may vary. Since the objective of the coverage function in this work is to select $N_{sel} = 3$ nodes to cover the target during each time step, the coverage function is defined as

$$B_{j,h}(\mathcal{N}_{B_{j,h}}(a)) = \begin{cases} b_1 \mathcal{N}_{B_{j,h}}(a) & \text{if } \mathcal{N}_{B_{j,h}}(a) \leq N_{sel} \\ b_1(6 - \mathcal{N}_{B_{j,h}}(a)) & \text{if } \mathcal{N}_{B_{j,h}}(a) > N_{sel} \end{cases} \tag{5.18}$$

where $\mathcal{N}_{B_{j,h}}(a)$ is the number of nodes that can the cell $\Omega_{v_{j,h}}$ with the joint action $a \in \mathcal{A}$, and b_1 is a design constant.

Based on the potential function defined in Eq. (5.16), the utility function for each player is computed based on the concept of *Marginal Contribution*.

$$\begin{aligned} \mathcal{U}(a_i, a_{-i}) &= \Phi(a_i, a_{-i}) - \Phi(\emptyset, a_{-i}) \\ &= \sum_{j=1}^{V} \sum_{h=1}^{V} v_{j,h}(B_{j,h}(a_i, a_{-i}) - B_{j,h}(\emptyset, a_{-i})) - \\ &\quad \frac{E_c(a_i) - E_c(\emptyset)}{|\mathcal{S}^*| E_c(R_L)} \end{aligned} \tag{5.19}$$

where $\emptyset$ represents player s_i's null action. In this work, it indicates the action R_0.

Proposition 5.5.1 *The game with potential function Φ of Eq. (5.16) and the utility function $\mathcal{U}_i$ of Eq. (5.19) is a potential game.*

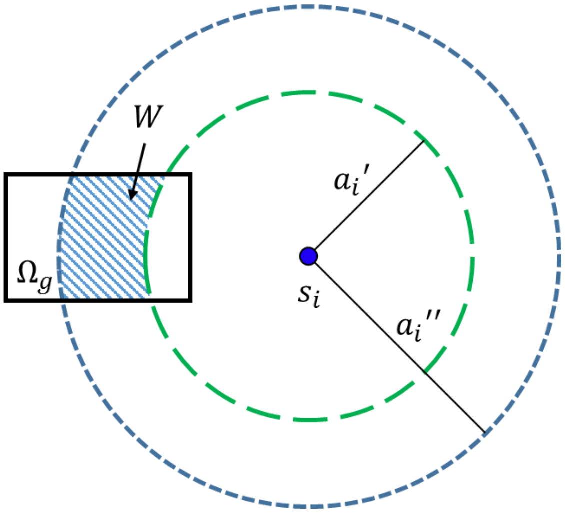


Figure 5.6: Illustrative Example for Lemma 1

Proof: Given a joint action a_{-i}, the difference in Φ for sensor node $s_i \in \mathcal{S}^\star$ to deviate its action from a_i' to a_i'' is:

$$\begin{aligned} &\Phi(a_i', a_{-i}) - \Phi(a_i'', a_{-i}) \\ &= (\Phi(a_i', a_{-i}) - \Phi(\emptyset, a_{-i})) - (\Phi(a_i'', a_{-i}) - \Phi(\emptyset, a_{-i})) \\ &= \mathcal{U}_i(a_i', a_{-i}) - \mathcal{U}_i(a_i'', a_{-i}) \end{aligned}$$

Thus game G satisfies Eq. (5.15) and it is a potential game. □

Coverage Function Design: In order to ensure that the game's equilibrium solution achieves a coverage degree of N_{sel}, the following conditions must be satisfied.

Lemma 1 *A node s_i will deviate from action a_i' to a_i'' if the coverage function $B_{j,h}(\mathcal{N}_{B_{j,h}}(a''))$ satisfies the following*

$$B_{j,h}(\mathcal{N}_{B_{j,h}}(a'')) - B_{j,h}(\mathcal{N}_{B_{j,h}}(a'') - 1) \geq \frac{E_c(a_i'') - E_c(a_i')}{W|\mathcal{S}^*|E_c(R_L)}, \tag{5.20}$$

where W is the sum of the worth gained by taking action a_i''.

Proof: Suppose that node s_i is considering deviating from action a_i' to a_i'' s.t. $a_i'' > a_i'$, as shown in Fig. 5.6. Let the set of cells in Ω_g covered by each joint action be $\Omega_{v_c}(a') \subset \Omega_{v_c}(a'') \subseteq \Omega_g$. Also, let the number of nodes covering the cells $\Omega_{v_{j,h}} \in \Omega_{v_c}(a')$ and $\Omega_{v_{j,h}} \in \Omega_{v_c}(a'')$ be a constant, $\mathcal{N}_B$, while the remaining cells are covered by $\mathcal{N}_B - 1$ nodes. Then, node s_i will choose the action a_i'' if

$$\mathcal{U}_i(a_i'', a_{-i}) - \mathcal{U}_i(a_i', a_{-i}) \geq 0. \tag{5.21}$$

Using Eq. (5.19), the above condition becomes

$$\sum_{j=1}^{V}\sum_{h=1}^{V} v_{j,h}(B_{j,h}(a'') - B_{j,h}(a')) - \frac{E_c(a_i'') - E_c(a_i')}{|\mathcal{S}^*|E_c(R_L)} \geq 0. \tag{5.22}$$

Then, for action a_i', the following target coverage is achieved

$$\begin{aligned}\sum_{j=1}^{V}\sum_{h=1}^{V} v_{j,h}B_{j,h}(a') &= B(\mathcal{N}_B) \sum_{\Omega_{v_{j,h}} \in \Omega_{v_c}(a_i')} v_{j,h}(\Omega_{v_{j,h}}) + \\ &\quad B(\mathcal{N}_B - 1) \sum_{\Omega_{v_{j,h}} \notin \Omega_{v_c}(a_i')} v_{j,h}(\Omega_{v_{j,h}}) \\ &= W'B(\mathcal{N}_B) + (1 - W')B(\mathcal{N}_B - 1), \end{aligned} \tag{5.23}$$

where W' is the sum of the worth in $\Omega_{v_c}(a')$. Similarly, action a_i'' results in the following target coverage

$$\sum_{j=1}^{V}\sum_{h=1}^{V} v_{j,h}B_{j,h}(a'') = W''B(\mathcal{N}_B) + (1 - W')B(\mathcal{N}_B - 1). \tag{5.24}$$

Finally, substituting Eq. (5.23) and (5.24) into Eq. (5.22) results in Eq. (5.20) as follows

$$\begin{aligned}(W'' - W')(B(\mathcal{N}_B) - B(\mathcal{N}_B - 1)) &\geq \frac{E_c(a_i'') - E_c(a_i')}{|\mathcal{S}^*|E_c(R_L)} \\ B(\mathcal{N}_B) - B(\mathcal{N}_B - 1) &\geq \frac{E_c(a_i'') - E_c(a_i')}{W|\mathcal{S}^*|E_c(R_L)}\end{aligned} \tag{5.25}$$

where $W = W'' - W'$. □

Remark 5.5.1 *Lemma 1 provides a criteria that allows the network designer to construct the coverage function in terms of the sum of the worth gained, W, between two actions.*

Theorem 5.5.1 *Given that the coverage function is already designed, the total coverage worth* *achieved by a node s_i's best action a_i^* is*

$$\sum_{j=1}^{V}\sum_{h=1}^{V} v_{j,h} \geq 1 - \frac{\Delta R}{\Delta B(a^*)|\mathcal{S}^*|R_L}, \tag{5.26}$$

where ΔR is the range difference between two consecutive actions, $\Delta B(a^*) = B_{j,h}(\mathcal{N}_{B_{j,h}}(a^*)) - B_{j,h}(\mathcal{N}_{B_{j,h}}(a^*) - 1)$, and $a^* = (a_i^*, a_{-i}^*)$.

Proof: Suppose that node s_i is trying to deviate from it's best action a_i^* which achieves a total coverage worth $\sum_{j=1}^{V}\sum_{h=1}^{V} v_{j,h} = W^*$. Additionally, suppose that node s_i is considering taking an action $a_i' > a_i^*$ or $a_i'' < a_i^*$. If action a_i' is considered, a total coverage worth W' is achieved. Then, plugging in a_i^* for a_i'' in Eq. (5.25), the following condition is achieved.

$$\begin{aligned}
(W^* - W')(B(\mathcal{N}_B) - B(\mathcal{N}_B - 1)) &\geq \frac{E_c(a_i^*) - E_c(a_i')}{|\mathcal{S}^*|E_c(R_L)} \\
(W^* - W')(\Delta B(a^*)) &\geq \frac{e_{HPS}^{s_i}\Delta T(a_i^* - a_i')}{|\mathcal{S}^*|e_{HPS}^{s_i}\Delta T R_L} \\
(W^* - W') &\geq \frac{(a_i^* - a_i')}{\Delta B(a^*)|\mathcal{S}^*|R_L} \\
W^* &\geq W' + \frac{(a_i^* - a_i')}{\Delta B(a^*)|\mathcal{S}^*|R_L}.
\end{aligned} \tag{5.27}$$

Here, $a_i^* - a_i' < 0$ and in order to maximize the right side of the Eq. (5.27), $a_i^* - a_i' = -\Delta R$ and $W' = 1$. Therefore, as long as the total coverage worth achieved by action a_i^* meets the following condition:

$$W^* \geq 1 - \frac{\Delta R}{\Delta B(a^*)|\mathcal{S}^*|R_L}, \tag{5.28}$$

then a_i^* is a better action than a_i'.

Similarly, if s_i is considering taking an action $a_i'' < a_i^*$, the following condition is derived from

Eq. (5.27)

$$\begin{aligned}(W^* - W'') &\geq \frac{(a_i^* - a_i'')}{\Delta B(a^*)|\mathcal{S}^*|R_L} \\ (W^* - W'') &\geq \frac{\Delta R}{\Delta B(a^*)|\mathcal{S}^*|R_L}.\end{aligned} \tag{5.29}$$

This states that the sum of the worth gained by taking action a_i^* must be greater than the right hand side of Eq. (5.29). Therefore, as long as W^* meets the condition of Eq. (5.27), action a_i^* is node s_i's best choice.

Therefore, the total worth achieved by a node s_i is bounded according to Eq. (5.27). □

Remark 5.5.2 *Theorem 5.5.1 provides a criteria that allows a node s_i to take an action that covers Ω_g. However, this design does not ensure that the coverage degree of the target's predicted position is N_{sel}.*

Theorem 5.5.2 *The coverage degree of the target's predicted position is N_{sel}, given that the coverage function is designed as follows:*

$$\begin{aligned}\Delta B(a) &\geq \frac{\Delta R}{W|\mathcal{S}^*|R_L}, \forall \mathcal{N}_{B_{j,h}}(a) \leq N_{sel} \\ \Delta B(a) &< \frac{\Delta R}{W|\mathcal{S}^*|R_L}, \forall \mathcal{N}_{B_{j,h}}(a) > N_{sel},\end{aligned} \tag{5.30}$$

where $\Delta B(a) = B_{j,h}(\mathcal{N}_{B_{j,h}}(a)) - B_{j,h}(\mathcal{N}_{B_{j,h}}(a) - 1)$.

Proof: Suppose that node s_i is considering deviating from action a_i' to a_i'' in the same manner as Lemma 1. Here, the goal of the game is to select N_{sel} nodes to cover the target's predicted location. Therefore, if there are less than N_{sel} nodes covering Ω_g, node s_i should be motivated to take action a_i''. However, if there are already N_{sel} or more nodes covering Ω_g, the node should not take the action a_i''. To ensure this occurs, the following conditions must be satisfied.

1. $\mathcal{U}_i(a_i'', a_{-i}) - \mathcal{U}_i(a_i', a_{-i}) \geq 0$, if $\mathcal{N}_{B_{j,h}}(a_i'') \leq N_{sel}$

2. $\mathcal{U}_i(a_i'', a_{-i}) - \mathcal{U}_i(a_i', a_{-i}) < 0$, if $\mathcal{N}_{B_{j,h}}(a_i'') > N_{sel}$

Using Lemma 1, the following conditions are derived.

$$B(a'') \geq B((a'') - 1) + b_2, if \mathcal{N}_{B_{j,h}}(a''_i) \leq N_{sel}, \tag{5.31}$$

$$B(a'') < B((a'') - 1) + b_2, if \mathcal{N}_{B_{j,h}}(a''_i) > N_{sel}. \tag{5.32}$$

where $b_2 = \frac{E_c(a''_i) - E_c(a'_i)}{(W'' - W')|\mathcal{S}^*|E_c(R_L)} = \frac{\Delta R}{W|\mathcal{S}^*|R_L}$ is the energy cost term. Therefore, as long as the coverage function satisfies the above conditions, the target's predicted position will achieve a coverage degree equal to N_{sel}. □

Remark 5.5.3 *Theorem 5.5.2 ensures that if a node s_i is trying to deviate from its current action a'_i to a''_i, it will only be motivated to select the action a''_i if $\mathcal{N}_{B_{j,h}}(a'') \leq N_{sel}$. Once $\mathcal{N}_{B_{j,h}}(a'') > N_{sel}$, the node will not be motivated to take the action a''_i. However, this does not guarantee that the resulting equilibrium will have N_{sel} nodes completely covering all of the cells in Ω_g. To achieve close the N_{sel} nodes completely covering all of the cells, $\Delta B(a)$ should be increased (or W should be decreased) until the desired condition in Theorem 5.5.1 is reached.*

• **Achieving the Optimal Equilibrium**: The sensor range selection game may contain more than one equilibrium, so it becomes imperative to employ an efficient learning algorithm to rapidly converge to the optimal equilibrium that can maximize the potential function Φ. For this purpose, the game leader s_L will utilize the Max-logit learning algorithm [109][110] to find the best joint action.

The Max-Logit algorithm adopts a repeated learning framework. At each iteration $m_g \in \mathbb{N}^+$, one player $s_i \in \mathcal{S}^\star$ is randomly selected and is allowed to alter its current action $a_i(m_g)$, while other players must repeat their actions, i.e., $a_{-i}(m_g)$. Then s_i randomly selects an alternative action $\hat{a}_i(m_g) \in \mathcal{A}_i$ with equal probability, and the associated utility $\mathcal{U}_i(\hat{a}_i(m_g), a_{-i}(m_g))$ is computed. Finally, it updates its action to $a_i(m_g + 1)$ in a probabilistic manner such that [109]:

$$a_i(m_g + 1) = \begin{cases} \hat{a}_i(m_g), & \text{with } Pr\big(a_i(m_g + 1) = \hat{a}_i(m_g)\big) = \mu \\ a_i(m_g), & \text{with } Pr\big(a_i(m_g + 1) = a_i(m_g)\big) = 1 - \mu \end{cases}$$

where $\mu = \frac{\psi(\hat{a_i})}{\max\{\psi(a_i),\psi(\hat{a_i})\}}$, $\psi(\hat{a_i}) = e^{\mathcal{U}_i(\hat{a_i},a_{-i})/\tau}$, and $\tau > 0$. The learning process stops when a predefined maximum learning steps $K \in \mathbb{N}^+$ is reached.

Upon reaching the equilibrium $a^\star \in \mathcal{A}$, the joint action is broadcast to each players and their sensing range is modified accordingly. This process outputs $\mathcal{R}^* = a^*$ and $\mathcal{S}^*$.

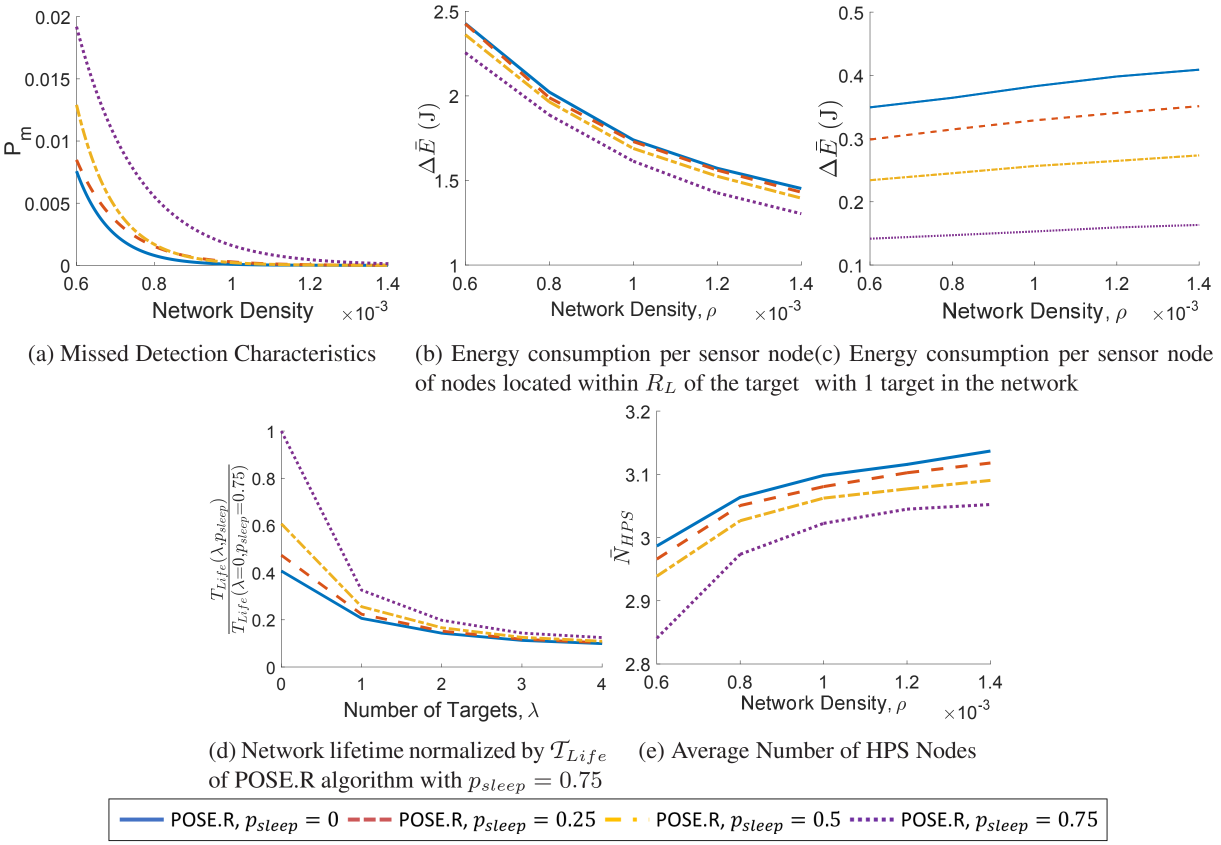
• ***Computation of the*** **HPS** ***State Transition Probability***: If $s_i \in \mathcal{S}^*$, then it has received or computed it's optimal sensing range, $R^{s_i}_{HPS}(k+1) \in \mathcal{R}^*$. Then, it computes the probability of the target traveling into the HPS devices coverage area, $P^{s_i}_{HPS}(k)$, according to Equation 3.12, however, the HPS sensing range is now $R^{s_i}_{HPS}(k+1)$. This probability is then used to update the state transition probabilities of the PFSA as described in Section 5.4.

5.6 POSE.R Algorithm Performance

To validate that the proposed algorithm facilitates resilient target coverage and tracking in the presence of coverage gaps, this section presents the POSE.R algorithm characteristics and compares them to existing methods. The POSE.R algorithm was simulated for 500 Monte Carlo runs in a $500m \times 500m$ Matlab environment with a single target traveling through the region. During each Monte Carlo run, the location of each sensor node is regenerated according to a uniform distribution. Each multi-modal sensor node is assumed to possess Passive Infrared (PIR) sensors as the LPS device and a laser range finder as the HPS device. The simulation energy costs, sensing ranges, process noises $(\sigma_{\boldsymbol{\upsilon},x}, \sigma_{\boldsymbol{\upsilon},y}, \sigma_{\boldsymbol{\upsilon},\psi})$, measurement noises (σ_R, σ_ϕ), and sensor selection parameters are presented in Table 5.1.

Table 5.1: Simulation Parameters

$e_{clock} = 0.01W$	$R_r = 15m$	$\alpha = 0.95$
$e_{LPS} = 115mW$	$R_{LPS} = 30m$	$\beta = 0.0036$
$e_{HPS} = 0.2\frac{W}{m}$	$R_c = 120m$	$N_{sel} = 3$
$e_{TX} = 1.26W$	$R_1 = 30m$	$N'_{sel} = 5$
$e_{RX} = 0.63W$	$R_L = 60m$	$\sigma_\phi = 0.25°$
$e_{DPU} = 1W$	$\Delta R = 6m$	$\sigma_R = 0.075m$
$E_0 = 137592J$	$\mu_{cl} = 0.025$	$\sigma_{\boldsymbol{\upsilon},x} = \sigma_{\boldsymbol{\upsilon},y} = 0.1m$
$\Delta T = 0.5s$	$p_{fa} = 0.01$	$\sigma_{\boldsymbol{\upsilon},\psi} = 0.1°$
	$\gamma_{HPS} = 0.95$	$W = 0.1$



(a) Missed Detection Characteristics

(b) Energy consumption per sensor node of nodes located within R_L of the target

(c) Energy consumption per sensor node with 1 target in the network

(d) Network lifetime normalized by $\mathcal{T}_{Life}$ of POSE.R algorithm with $p_{sleep} = 0.75$

(e) Average Number of HPS Nodes

Figure 5.7: POSE.R Network Characteristics

5.6.1 POSE.R Characteristics

This subsection presents the POSE.R algorithm characteristics to validate the control requirements of this thesis. Additionally, the POSE.R characteristics are presented to determine the design parameters ρ, p_{sleep}, and N'_{sel}.

5.6.2 Missed Detection and Energy Characteristics

First, the POSE.R algorithm's missed detection and energy consumption characteristics are presented in Fig. 5.7. To understand the effects of the network density, ρ, and sleeping probability, p_{sleep} parameters, the POSE.R algorithm was simulated with a combination of the following parameters: $\rho = [0.6e^{-3}, 1.4e^{-3}](\frac{nodes}{m^2})$ and $p_{sleep} = [0, 0.25, 0.5, 0.75]$. Figure 5.7a presents the probability of missed detection, P_m, vs network density for various p_{sleep} values. This result shows that for a low density network, the POSE.R algorithm is able to obtain a $P_m < 0.02$ while for medium to high density networks, the algorithm is able to achieve a $P_m < 0.002$. This indicates that the POSE.R algorithm is able to achieve low missed detection rates by selecting at least 1 node to be in the HPS state around the target's predicted position, thus enabling opportunistic sensing.

The energy characteristics of the POSE.R algorithm are presented in Figures 5.7b and 5.7c. In Fig. 5.7b, the average energy consumption per sensor node located within a distance of R_L of the target's location is presented for various network densities and p_{sleep} values. This result shows that as the network density increases, the average energy consumption per sensor node decreases. This is because for low density networks, the base coverage degree is insufficient, i.e. $D_{R_1}(\mathbf{u}^{\tau_\ell}) < N_{sel}$, and the adaptive sensor selection method triggers a game to adjust the selected nodes HPS range. This causes the selected nodes to increase their energy consumption to ensure target coverage, thus adding resilience to the network. Additionally, the nodes not selected that are located a distance $> R_1$ away from the target are more likely to be in the LPS state to maintain awareness. For high density networks, the selected nodes can cover the target with a sensing range $R^{s_i}_{HPS}(k) = R_1$ which reduces each nodes energy consumption. The nodes not selected transition between the *Sleep* and *LPS* state based on the LPS detection model which further reduces the average energy consumption.

Additionally, Fig. 5.7c presents the average energy consumption per sensor node located a distance $> R_L$ away from the target's location. This shows that the average energy consumption slightly increases with density but is mainly affected by the parameter p_{sleep}. This is because the nodes located away from the target are cycling between the low power consuming states (*Sleep* and *LPS*) where the p_{sleep} parameter controls the frequency of each node being in the *Sleep* state. Additionally, this shows that the nodes away from the target are very energy efficient and are only utilizing their energy reserves when a target is located within a distance R_L of the nodes, thus opportunistically sensing.

Finally, Fig. 5.7d presents the network lifetime defined in Definition 4.5.1. The network lifetime is computed over a tube with radius R_{LPS} because once the nodes within the tube die, a new target will never be detected and initialized by nodes transitioning between the *Sleep* and *LPS* states. Therefore, the POSE.R algorithm was simulated in a $2R_L \times 600m$ tube with targets traveling through the center of the tube in a straight line. The number of targets, λ, located within the

Table 5.2: Energy Geometric Dilution Of Precision (EGDOP) Measures

	Energy Distribution among sensor nodes					
	$[0.5E_0, E_0]$	$[0.6E_0, E_0]$	$[0.7E_0, E_0]$	$[0.8E_0, E_0]$	$[0.9E_0, E_0]$	$[E_0, E_0]$
$\bar{E}_{\%,increase}$ **(%)**	7.301	5.692	3.920	2.482	1.177	0.004
$\bar{E}_{eff}(EGDOP)$	0.962	0.968	0.977	0.985	0.992	1.00
$\bar{E}_{eff}(GDOP)$	0.933	0.946	0.962	0.976	0.988	1.00
$D_{KL}(GDOP\|\|EGDOP)$	0.0315	0.0317	0.0309	0.0307	0.0309	0.0424
$D_{KL}(GDOP\|\|MaxEnergy)$	0.3190	0.3278	0.3205	0.3154	0.3138	0.4093

tube during every time step is varied between $\lambda = [0, 3]$ and the network lifetime is presented in Fig. 5.7d for various values of p_{sleep}. As seen, p_{sleep} drastically changes the network lifetime when $\lambda < 1$ targets are within the tube at the same time. This is because the nodes are mainly cycling between *Sleep* and *LPS* states and consuming energy similar to Fig. 5.7c. However, as λ increases, the number of nodes within R_L of the targets becomes all of the nodes in the tube, resulting in the nodes consuming energy similar to Fig. 5.7b. Therefore, utilizing the missed detection, energy, and network lifetime characteristics, the network designer can choose the appropriate values of the parameters ρ and p_{sleep} to meet their requirements.

5.6.3 Adaptive Sensor Selection Characteristics

Next, the Adaptive Sensor Selection characteristics are presented to show the benefits of EGDOP and potential games. First, the average number of sensor nodes in the *HPS* state are presented in Fig. 5.7e. This result shows that $N_{sel} = 3$ nodes are selected to track the target during each time step which allows for a coverage degree $D(\mathbf{u}^{\tau_\ell}) = N_{sel}$. As seen in Fig. 5.7e, for network densities $\rho \geq 0.8e^{-3}$, the number of *HPS* nodes is slightly larger than N_{sel}. This is due to the false alarm probability p_{fa} causing sensors away from the target to transition to the *HPS* state. This effect is minimized for higher p_{sleep} values since the sensors away from the target are more likely to be in the *Sleep* state. For low network densities $\rho < 0.8e^{-3}$, the number of *HPS* state sensors is slightly below N_{sel}. This is because sensor nodes along the targets path may get stuck in the *Sleep* state for values of $p_{sleep} > 0$. Therefore, for low density networks, the number of candidates for sensor selection may be $< N_{sel}$. Based on the missed detection, energy consumption, and number of *HPS* state nodes characteristics, the parameters $\rho = 1.4e^{-3}$ and $p_{sleep} = 0.5$ were chosen to ensure low missed detection rates and energy consumption.

Next, the properties of the EGDOP selection method are studied in Table 5.2 by comparing EGDOP selection with GDOP selection and Max Energy selection. The GDOP selection method [29] selects N_{sel} sensors from the set $\mathcal{S}_{det}$ that are geometrically diverse and minimize the predicted state covariance error. The Max Energy selection method selects N_{sel} sensors from the set $\mathcal{S}_{det}$ that have the maximum energy remaining. To validate that EGDOP selection selects nodes that are geometrically diverse and energy efficient, each selection method is simulated with a varying initial energy distribution. Here, each sensor is initialized with a random initial energy according to a uniform distribution. This simulates the effects of a network being deployed for a long period of time where targets have passed through the network and each node possess a different remaining energy. The columns of Table 5.2 represent the lower and upper bounds of the uniform distribution used to initialize each nodes energy, while the rows of correspond to the following:

- Average Percent Energy Increase:

$$\bar{E}_{\%,increase} = \sum_{s_i \in \mathcal{S}^*} \hat{E}_R^{s_i}(k) - \sum_{s_i \in \mathcal{S}^*_{GDOP}} \hat{E}_R^{s_i}(k), \tag{5.33}$$

  where $\hat{E}_R^{s_i}(k) = \frac{E_0 - E^{s_i}(k) - \omega R_1 \Delta T}{E_0}$ and $\mathcal{S}^*_{GDOP}$ consists of N_{sel} nodes selected using GDOP selection.

- Average EGDOP Energy Efficiency:

$$\bar{E}_{eff}(EGDOP) = \frac{\sum_{s_i \in \mathcal{S}^*} \hat{E}_R^{s_i}(k)}{\sum_{s_i \in \mathcal{S}^*_{MaxEnergy}} \hat{E}_R^{s_i}(k)}, \tag{5.34}$$

  where $\mathcal{S}^*_{MaxEnergy}$ consists of N_{sel} nodes selected using Max Energy selection.

- Average GDOP Energy Efficiency:

$$\bar{E}_{eff}(GDOP) = \frac{\sum_{s_i \in \mathcal{S}^*_{GDOP}} \hat{E}_R^{s_i}(k)}{\sum_{s_i \in \mathcal{S}^*_{MaxEnergy}} \hat{E}_R^{s_i}(k)}, \tag{5.35}$$

- Amount of information lost when the predicted state distribution $\mathcal{N}(\dagger)$ achieved using $\dagger$ sensor selection is used to approximate the predicted state distribution $\mathcal{N}(\dagger')$ achieved using

$\dagger'$ sensor selection (Kullback-Leibler Divergence):

$$D_{KL}(\mathcal{N}(\dagger)||\mathcal{N}(\dagger')) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathcal{N}(\dagger) log(\frac{\mathcal{N}(\dagger)}{\mathcal{N}(\dagger')})dydx \tag{5.36}$$

where $\mathcal{N}(\dagger) = \mathcal{N}(H\hat{\mathbf{x}}^{s_i}(k+1|k), \hat{\mathbf{\Sigma}}(\dagger))$, $\mathcal{N}(\dagger') = \mathcal{N}(H\hat{\mathbf{x}}^{s_i}(k+1|k), \hat{\mathbf{\Sigma}}(\dagger'))$, and

$$\mathbf{\Sigma}(\dagger) = (\sum_{s_h \in \dagger} \frac{1}{\sigma_\theta^2 r_{s_h}^2} \begin{bmatrix} sin^2(\phi_{s_h}) & -sin(\phi_{s_h})cos(\phi_{s_h}) \\ -sin(\phi_{s_h})cos(\phi_{s_h}) & cos^2(\phi_{s_h}) \end{bmatrix})^{-1} \tag{5.37}$$

is the state covariance error achieved with the set of sensors, $\dagger$, and no prior information; σ_θ is the measurement angle standard deviation; r_{s_h} is the range between node s_h and the target; and ϕ_{s_h} is the angle between node s_h and the target.

As seen in Table 5.2, the average percent energy increase of EGDOP vs GDOP in always positive and increases as the variation of initial energy increases. This shows that by incorporating energy into the cost function, the EGDOP selection method selects nodes with a higher energy remaining than GDOP selection. Then, comparing the energy efficiency of EGDOP vs. Max Energy and GDOP vs. Max Energy, EGDOP is able to achieve a higher energy efficiency than GDOP. This further justifies the benefits of adding energy remaining into the cost function.

The Kullback-Leibler (KL) divergence is presented to justify that the sensors selected using EGDOP are geometrically diverse. The geometric diversity is measured based on the predicted state covariance error. When the covariance error is minimized, the sensors selected are geometrically diverse. This is because sensors that are geometrically diverse create a circular covariance error while sensor that are co-located create an elliptical error with more uncertainty. Since the GDOP selection method minimizes the covariance error, the EGDOP selection method should provide a better approximation of the covariance error than the Max Energy selection method. Therefore, the results presented in rows 4 and 5 of Table 5.2 show that the KL divergence from EGDOP to GDOP is much smaller than the divergence from Max Energy to GDOP. This indicates that EGDOP is a better approximation of the predicted state covariance error than Max Energy. Additionally, the values in row 4 of Table 5.2 are very small which indicates that the predicted co-

Table 5.3: Game Results

			N'_{sel}		
	3	4	5	6	7
$\sum_{j,h}^{V} \sum_{j,h}^{V} v_{j,h}$ (%)	99.81	99.95	99.96	99.97	99.98
$\bar{N}(a^* \geq R_1)$	2.999	3.005	3.015	3.036	3.056
$\bar{\phi}_{increase}$ (%)	48.9	51.7	52.7	51.7	55.4
$\frac{\phi_{Game}}{\phi_{Opt}}$	0.999	0.980	0.976	0.974	0.973
$t_{Game}(s)$	0.050	0.065	0.079	0.098	0.129
$t_{Opt}(s)$	0.088	0.808	8.137	56.300	517.957

variance error achieve using EGDOP selection is a very good approximation for the predicted covariance error of GDOP selection. Therefore, the results presented in Table 5.2 show that EGDOP selection selects nodes that are geometrically diverse and energy efficient.

Finally, the potential game characteristics are presented in Table 5.3. Here, the number of players, N'_{sel}, were varied between $N'_{sel} = [3, 7]$ and the following measures are presented.

- Average coverage worth achieved by the equilibrium: $\sum_{j,h}^{V} \sum_{j,h}^{V} v_{j,h}$,
- Average Number of sensors whose action is greater than or equal to R_1: $\bar{N}(a^* \geq R_1)$,
- Average potential function gain: $\bar{\phi}_{increase} = \phi(a^*) - \phi(a_0)$, where a_0 is the initial joint action,
- Potential function efficiency: $\frac{\phi_{Game}}{\phi_{Opt}} = \frac{\phi(a^*)}{\phi(a^*_{opt})}$, where a^*_{opt} is the globally optimal joint action achieved through exhaustive search,
- Potential Game Computation Time: t_{Game},
- Exhaustive Search Computation Time: t_{Opt}.

The first two rows of Table 5.3 are presented to validate Theorem 5.5.1 and Theorem 5.5.2. Here, the average sum of the worth achieved by the game exceeds the lower bound of Theorem 5.5.1 and the average coverage degree achieved is equal to N_{sel}, thus validating the properties presented in Theorem 5.5.1 and Theorem 5.5.2. Additionally, row 3 in Table 5.3 shows that the potential function always increases from the initial joint action, thus satisfying the property of a potential game. Furthermore, the game efficiency as compared to the optimal solution, shown in row 4 of Table 5.3, is very close to 1. This indicates that the Maxlogit learning algorithm is very efficient in searching for the Nash equilibrium. Finally, the computation time between the game and the exhaustive search are presented in rows 5 and 6 of Table 5.3. This shows that the game significantly decreases the computation time, making the sensor range selection game practical for real-time implementation. Once $N'_{sel} > 4$ the computation time of the exhaustive search becomes impractical for real-time implementation which further validates the use of a potential game to select the nodes optimal sensing ranges.

5.6.4 POSE.R vs. Existing Techniques

Next, the POSE.R algorithm is compared to existing scheduling methods. Specifically, POSE.R is compared against three fixed range distributed scheduling methods. Since the varying range methods presented in the related work section do not consider target tracking, POSE.R is not compared against them. The methods compared against are as follows: (1) Autonomous Node Selection (ANS), (2) LPS-HPS Scheduling, and (3) Random Scheduling. The ANS algorithm [29] is a distributed sensor selection method that utilizes GDOP to identify the best sensors to track the target. Here, the sensors collaborate in a distributed manner to make scheduling decisions. However, the ANS algorithm is presented for passive sensors and does not consider multi-modal sensor nodes. Therefore, to ensure an apple to apple comparison, the ANS algorithm is adapted to include multi-modal operating conditions. The ANS algorithm has the sensors selected actively sensing and tracking the target, while the sensors not selected are in a low power state with their receiver on. Therefore, the nodes are adapted to switch between the *LPS* (passive) and *HPS* (active) states to mimic the behavior of the ANS algorithm. Thus, if a sensor selects itself to track the target, it will transition to the *HPS* state, while staying in the *LPS* state to keep the receiver enabled and initialize tracking when not selected.

The LPS-HPS Scheduling method is a distributed trigger-based activation method that utilizes two operating states, passive (*LPS*) and active (*HPS*). The nodes remain in the passive state until a target is detected. Once a target is detected, the node remains in the active state until the target passes out of the node's field of view. The Random Scheduling method is a distributed probabilistic method where the sensors randomly switch between sleeping and actively sensing (*HPS*). Here, during each time step, a node sleeps with a probability p_{rand} and senses with a probability $1 -$

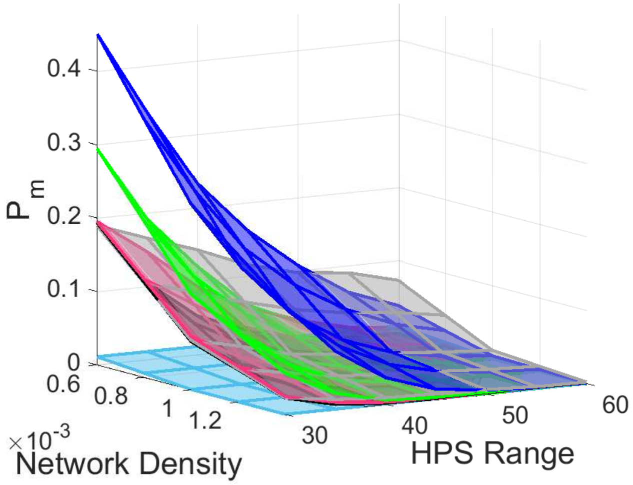


Figure 5.8: Missed Detection Rates Comparison

p_{rand}. Thus, for $p_{rand} = 1$ the network is always sleeping while for $p_{rand} = 0$, the network is always sensing. Note that the LPS-HPS and Random Scheduling methods do not facilitate sensor collaboration.

Missed Detection and Root Mean Squared Error Performance: First, the missed detection characteristics of POSE.R are compared to the other distributed scheduling methods shown in Fig. 5.8. Here, each of the other scheduling methods were simulated with a fixed sensing range between $R^{s_i}_{HPS}(k) = [R_1, R_L]$ and are shown in Fig. 5.8. As seen in Fig. 5.8, the POSE.R algorithm achieves a significantly lower missed detection probability than the other methods for low HPS sensing ranges. As the network density and the HPS sensing range increases, the P_m for the other distributed methods approaches the POSE.R algorithm. Therefore, in order for the other methods to achieve similar characteristics as POSE.R, the network must contain a high density of sensor nodes that are utilizing a large HPS sensing range.

Next, the tracking performance of the POSE.R algorithm and other distributed methods is presented in Figures 5.9 and 5.10. As seen, the POSE.R and ANS algorithms achieve significantly lower position and velocity Root Mean Squared Errors (RMSE) as compared to the other methods. This is due to distributed sensor collaboration that allows the nodes to perform fusion to reduce the covariance error. Also, the sensor selection cost function in POSE.R and ANS aims to select the set of sensors that minimizes the measurement covariance error. Therefore, the performance of

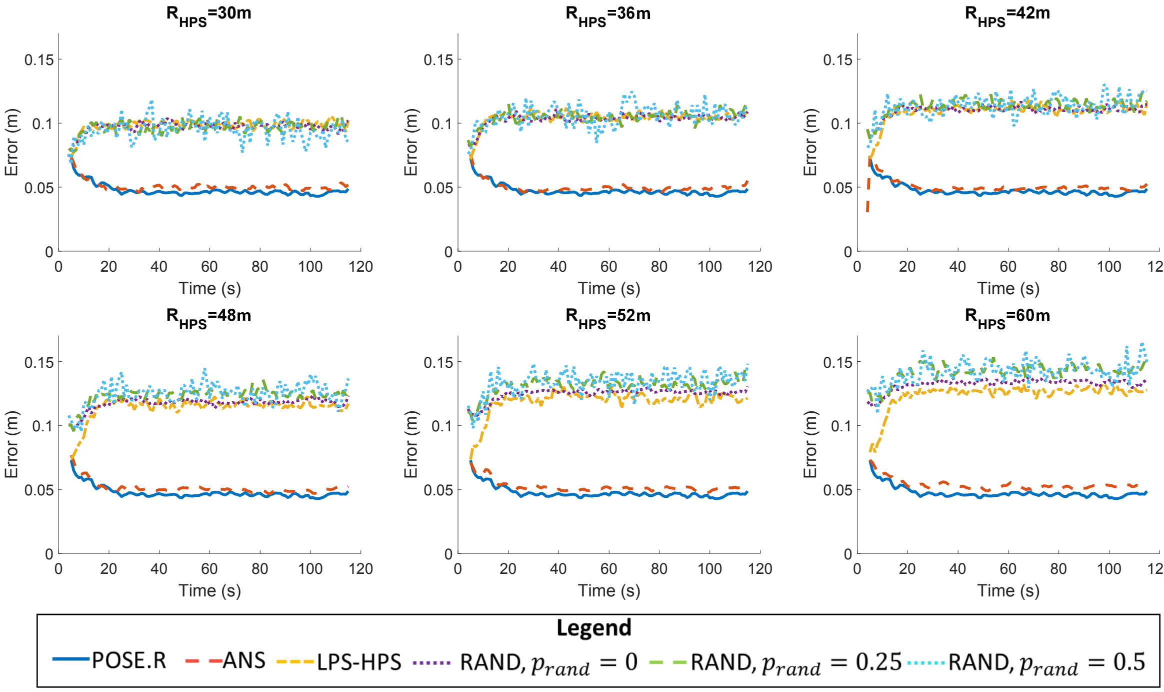


Figure 5.9: POSE.R algorithm Position Root Mean Squared Error

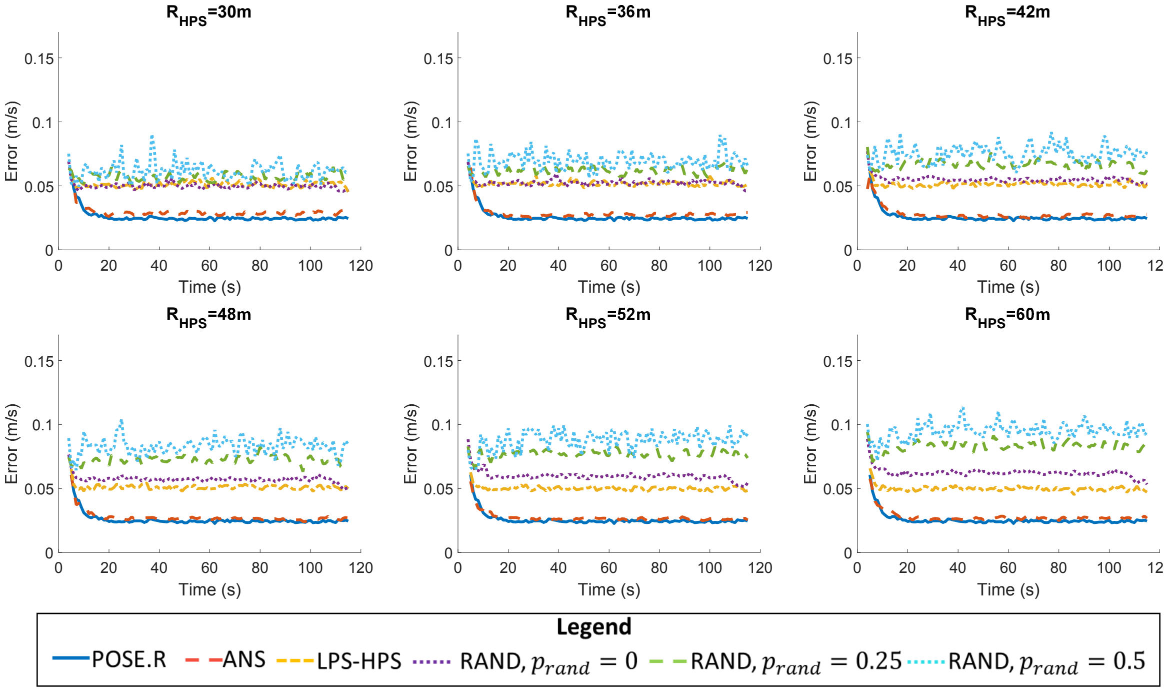
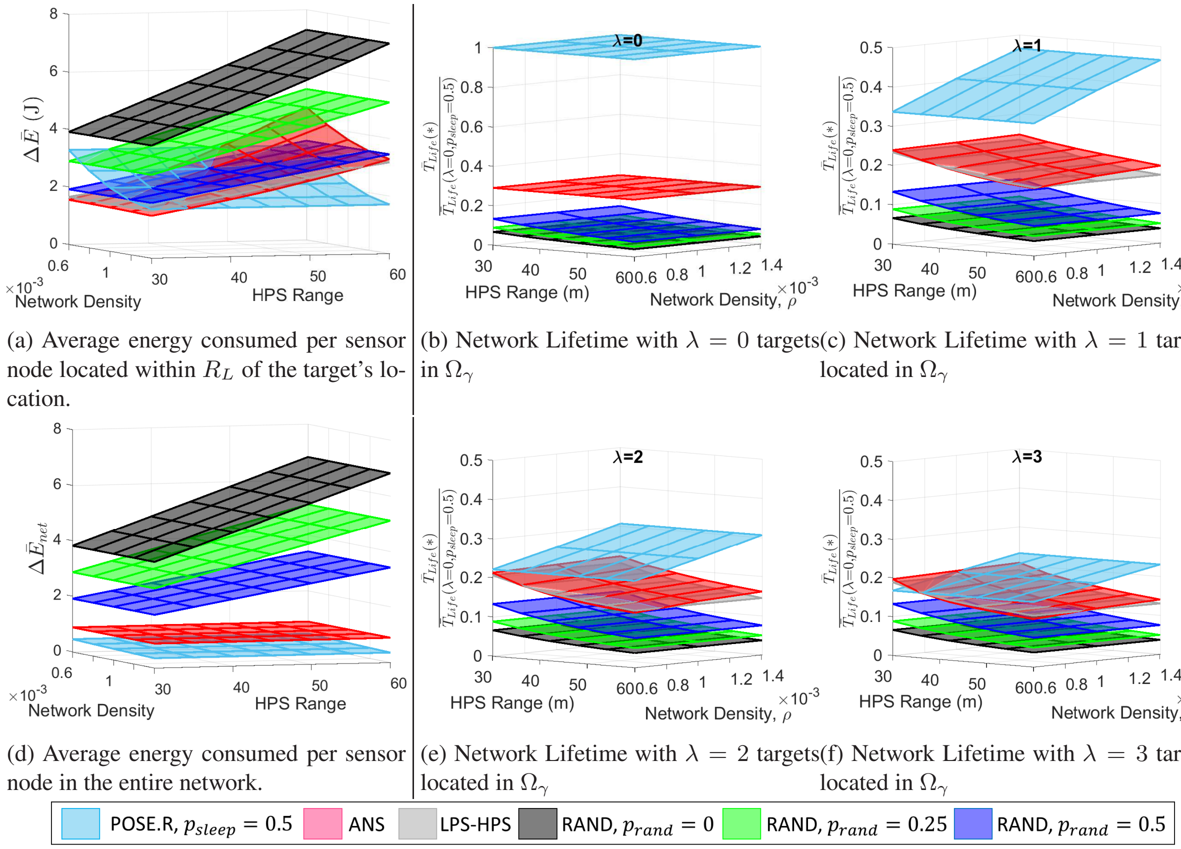


Figure 5.10: POSE.R algorithm Velocity Root Mean Squared Error

the POSE.R and ANS algorithms achieve similar results. Additionally, it can be seen that as the HPS sensing range increases, the RMSE of the LPS-HPS and Random methods slightly increases.



(a) Average energy consumed per sensor node located within R_L of the target's location.

(b) Network Lifetime with $\lambda = 0$ targets in Ω_γ

(c) Network Lifetime with $\lambda = 1$ targets located in Ω_γ

(d) Average energy consumed per sensor node in the entire network.

(e) Network Lifetime with $\lambda = 2$ targets located in Ω_γ

(f) Network Lifetime with $\lambda = 3$ targets located in Ω_γ

Figure 5.11: POSE.R algorithm energy characteristics compared to existing techniques.

This is because nodes further away from the target are able to receive measurements. As the HPS range increases, the measurement noise increases, causing the nodes further away from the target to have an increased error.

Energy Consumption Performance: The above results indicate that through efficient design, the other distributed networks can achieve the same missed detection characteristics as POSE.R. However, if the energy consumption characteristics are compared, the opposite result is achieved. Specifically, the average energy consumption per sensor node located within a distance of R_L of the target's location is presented in Fig. 5.11a. This result shows that for low densities and low HPS ranges, the POSE.R network consumes more energy compared to the other distributed methods. This is because for low densities, $D_{R_1}(\mathbf{u}^{\tau_\ell}) < N_{sel}$, which requires the selected nodes to expand their sensing ranges; while the other distributed methods utilize a smaller sensing range which consumes less energy. However, as the network density increases for lower HPS ranges, the POSE.R algorithm energy consumption approaches the ANS algorithms energy consumption.

This is because within R_L of the network, N_{sel} sensors are selected to be in the *HPS* state and the remaining are likely to be within the *LPS* state. This is the same characteristics of the ANS network which is seen in Fig. 5.11a. Then, as the HPS range increases, the energy consumption of the other network increases above the POSE.R energy consumption. This is because the POSE.R algorithm can select a smaller sensing range to cover the target. This shows the true benefits of the adaptive sensor selection algorithm. Therefore, if the network is designed based on energy consumption, the POSE.R will achieve a lower P_m than the other networks. However, if the network is designed based on P_m characteristics, the POSE.R algorithm will consume less energy per sensor node which increases the network lifetime.

Additionally, the multi-modal sensor node design benefits are presented in Fig. 5.11d in terms of the average energy consumption per sensor node located a distance greater than R_L of the targets location. This result shows that by incorporating a *Sleep* state within the distributed supervisor, the POSE.R algorithm is able to consume less energy than all of the other networks. This result also shows that the POSE.R, LPS-HPS, and ANS algorithms are opportunistic sensing methods. When comparing Fig. 5.11a with Fig. 5.11d, these methods shows that the nodes located away from an event consume significantly less energy than the nodes located around the event. This indicates that these scheduling methods opportunistically utilize their energy resources.

Next, the network lifetime of the POSE.R network is compared against the other distributed scheduling methods in Figures 5.11b, 5.11c, 5.11e, 5.11f. Here, each network was simulated with a network density $\rho = 1.4e^{-3}$ in a $2R_L \times 600m$ tube with λ targets traveling through the center of the tube in a straight line. As a target leaves the tube, a new target is generated at the beginning of the tube to ensure that λ targets are located within the tube during each time step. First, the network lifetime with $\lambda = 0$ targets within the network is presented in Fig. 5.11b. Here, the POSE.R algorithm has a significantly longer network lifetime as compared to the other methods. This is because all of the nodes in the network are consuming energy at the rate presented in Fig. 5.11d. This result shows that by incorporating multi-modal sensor nodes into the DSN, the network lifetime is significantly increased. Next, the network lifetime with $\lambda \geq 1$ targets located within the network at each time step is presented in Figures 5.11c, 5.11e, 5.11f. These results show that the POSE.R algorithm has an increased network lifetime even when targets are present. However, as λ increases, the POSE.R algorithm network lifetime does not show significantly large lifetime benefits. This is because the tube Ω_γ becomes completely occupied with target's and all of the nodes begin to consume energy at the rate presented in Fig. 5.11a. For low densities and low HPS ranges, the POSE.R algorithm does not have an increased lifetime as compared to ANS. This because the adaptive sensor selection algorithm increases the HPS range of the selected sensors to ensure target coverage, while ANS may not be able to ensure target coverage with a low HPS range.

When comparing the results presented in Figures 5.8, 5.9, 5.10, and 5.11, it can be seen that the POSE.R algorithm outperforms the other distributed scheduling methods. Although the P_m achieved becomes the same for high densities and high HPS ranges, the energy characteristics, RMS errors, and network lifetimes achieved are significantly lower for POSE.R.

Network Resilience to Coverage Gaps: Finally, the resilience of the POSE.R network was analyzed by evaluating the algorithm's performance in the presence of a coverage gap, as shows in Figures 5.12 and 5.13. Here, each network was simulated with a network density of $\rho = 1.4e^{-3}$ and a single target travels through the ROI. However, the nodes located within a distance of $R_{gap} = [30, 50]$ of the target's location at time $k = 50s$, $\mathbf{u}^{\tau_\ell}(50)$, is initialized with an initial energy value $E_0 = 0$. This simulates a group of sensor nodes dying and a coverage gap of size $>= R_{gap}$. As seen in Fig. 5.12, the probability of detection, P_{det}, is presented verses time for various R_{gap} and $R^{s_i}_{HPS}(k)$ values. This result indicates that for low HPS ranges, the other methods lose the target when it travels through the coverage gap. However, the POSE.R algorithm is able to cover the target by performing adaptive sensor selection and optimal range selection. Additionally, similar to Fig. 5.8, as the HPS range increases, the P_{det} achieved with the ANS and Random scheduling methods are the same as the POSE.R algorithm.

Then, the average energy consumption per sensor node located within R_L of the target vs. time is presented in Fig. 5.13. This result shows that as the target travels through the coverage gap, the POSE.R algorithm's average energy consumption increases since the nodes within R_L of the target must expand their sensing range to achieve target coverage. Additionally, as the HPS range

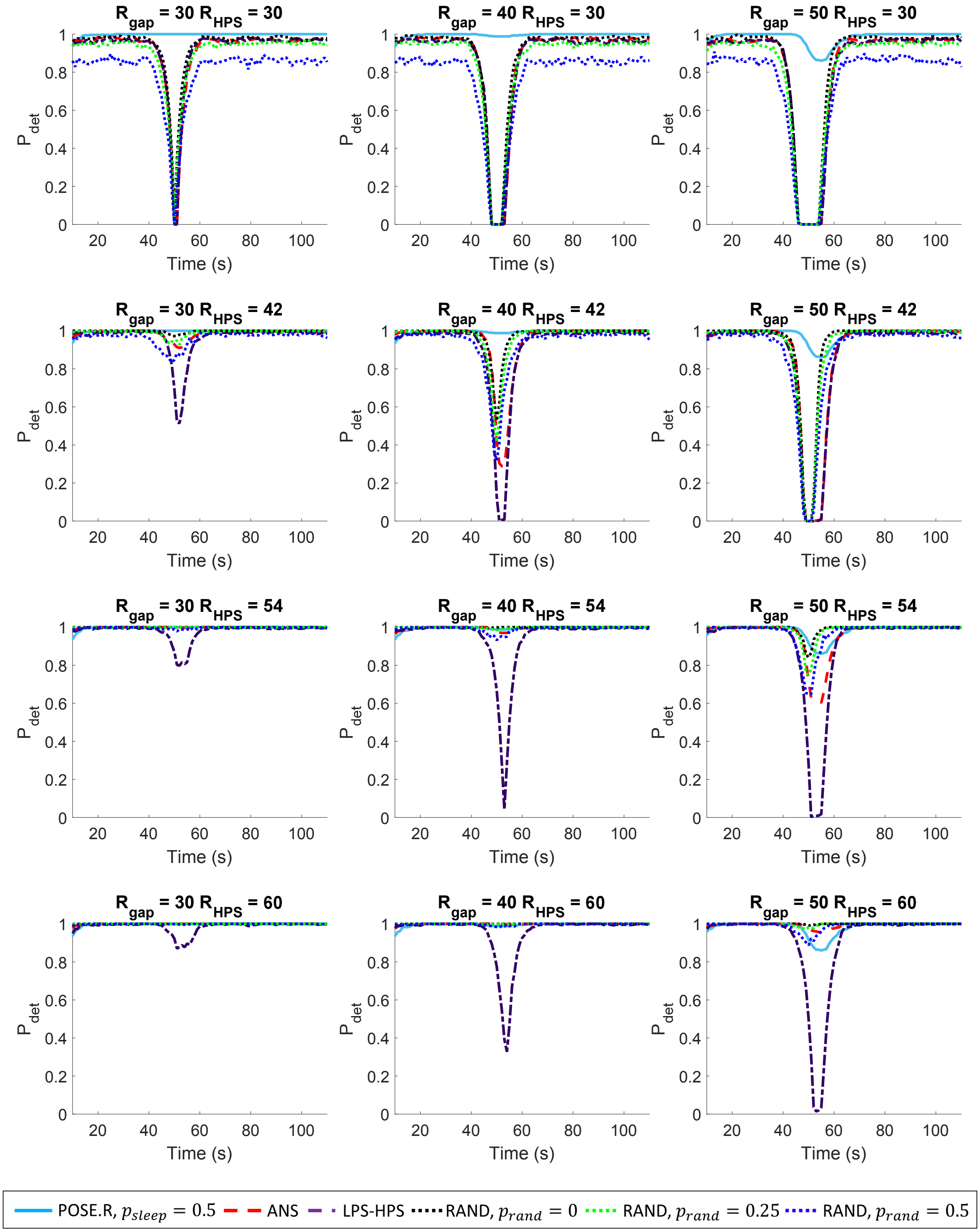


Figure 5.12: Probability of detection vs. time as a target travels through a coverage gap located at $t = 50s$.

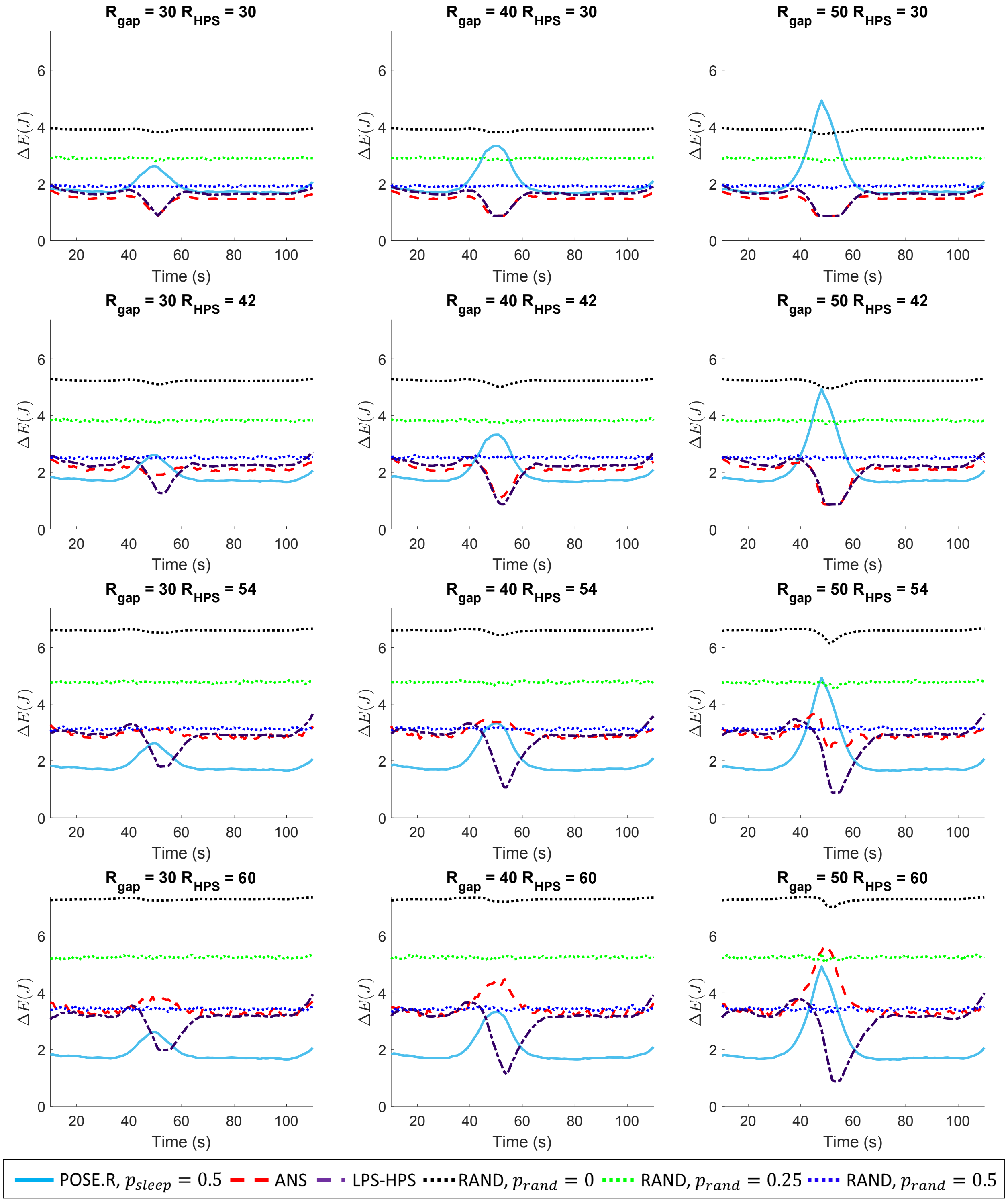


Figure 5.13: Average energy consumed per sensor node within R_L of the the target. The target travels through a coverage gap located at $t = 50s$.

increases, the POSE.R algorithm energy savings increases which further shows the benefits of the multi-modal control algorithm. Additionally, when achieving the same P_{det} as the other networks, the POSE.R algorithms average energy consumption is less than the other networks. Therefore, Figures 5.12 and 5.13 show that the POSE.R algorithm is resilient to coverage gaps by performing optimal range selection, as well as energy efficient as compared to existing distributed methods.

5.7 Conclusion

This chapter developed the POSE.R algorithm for controlling a distributed sensor network for resilient energy-efficient target tracking. The network control approach consists of detecting, tracking, and collaborating target state information in a distributed manner. Then, the target information is predicted and utilizes the select the best sensors to opportunistically cluster around a target to enhance target tracking. Additionally, the optimal sensing range is selected for the best sensors to ensure target coverage, incorporate resilience in the event of a covarage gap, and minimize energy consumption. The multi-modal states of each sensor node are probabilistically controlled using distributed PFSA-based supervisors. The POSE.R network characteristics were compared with existing distributed scheduling methods and the results show that the POSE.R algorithm extends the networks lifetime, incorporates network resiliency, achieves high tracking accuracy, and low missed detection rates.

CHAPTER 6

CONCLUSIONS

This thesis presents a DSN control algorithm to solve the *Target Coverage* problem. The objective of this thesis is to maximize the network lifetime while ensuring that every target is covered by at least 1 active sensor node. To maximize the network lifetime, the sensor nodes must perform their desired task, e.g. target tracking, for long periods of time. However, each node only possess a finite energy resource, which once depleted, causes the node to die and decreases the network lifetime. To ensure that all of the targets are covered by at least 1 sensor node, the sensor nodes must adapt to the network density around the target's predicted position to provide resilient target coverage. Therefore, this thesis develops an energy-efficient node-level control strategy that extends the network lifetime, while providing resilient target coverage in the presence of sensor failures. The first two themes of this thesis tackle the energy-efficiency objective, while the third theme addresses resilient target coverage for a stationary DSN.

6.1 Prediction-based Opportunistic Sensing (POSE) algorithm

The first theme of this thesis address the problem of energy-efficiency in DSN by developing the *Prediction-based Opportunistic Sensing* (POSE) algorithm. This algorithm manages the power consumption of each sensor node in a distributed fashion to enable energy-efficient target coverage. This is achieved by embedding a distributed supervisor, i.e. a *Probabilistic Finite State Automaton*

(PFSA) on each node, which enables and disables the devices on the node in an opportunistic manner to conserve energy.

The states of the PFSA consist of three operating conditions that determine the devices enabled on each sensor node and are: 1) *Sleep*, 2) *Low Power Sensing* (*LPS*), and 3) *High Power Sensing* (*HPS*). The *Sleep* state consumes minimal energy by disabling all devices. The *LPS* state allows the node to detect the target using LPS devices or from information transmitted by its neighbors. The *HPS* state allows the node to track the target and broadcast target information to its neighbor to alert them of the target locations. The state transition probabilities of the PFSA are dynamically updated based on the target locations. The goal of the PFSA design is to only enable the *HPS* state when a target is predicted to be located in the nodes coverage area, thus opportunistically sensing. Additionally, when a target is located away from the node, the node should transition between the *Sleep* and *LPS* states to minimize energy consumption.

In order for the node to transition to the *HPS* state, it must predict with high confidence that a target is located within its coverage area. This can only be achieved in the *HPS* and *LPS* states. Thus, only activating *HPS* sensors around the target locations significantly increases the energy-efficiency of the network. Additionally, allowing for a low power target detection state, i.e. *LPS*, the network can periodically check to identify if a target is present while conserving energy. Furthermore, by incorporating a *Sleep* state, the sensor node can further reduce energy consumption by turning everything off. This results in an extended network lifetime and is the first objective of the target coverage problem

Therefore, the first theme of this thesis provides the following novel contributions:

- The development of distributed supervisors (i.e. PFSA) for probabilistic control of the devices enabled on each multi-modal sensor node, and
- A prediction-based scheduling approach that updates the probabilities of the distributed supervisor to enable opportunistic sensing for energy-efficient control.

The POSE algorithm was simulated and compared against two distributed scheduling approaches. First, the missed detection characteristics were presented and showed that the POSE algorithm almost achieves the same probability of missed detection as an always monitoring network. This result shows that even though the POSE algorithm incorporates three operating conditions, the design of the PFSA and the algorithms embedded within each state still allow each node to be aware of the target as it travels through the network. This is achieved by incorporating prediction-based opportunistic sensing into the network. The sensor nodes are able to identify the target locations and ensure that they transition to the *HPS* state along the target's path to minimize missed detections.

Next, the POSE algorithm energy savings compared to the two distributed scheduling approaches was presented. This result showed that the POSE algorithm achieves $\geq 90\%$ energy savings compared to a network that is always monitoring. Additionally, the POSE algorithm achieves $\geq 80\%$ energy savings compared to Random Scheduling networks and $\geq 30\%$ against Trigger-based activation networks. This shows that designing an intelligent multi-modal sensor node control strategy, the energy-efficiency can be significantly increased which leads to an extended network lifetime.

Finally, the tracking performance of the POSE algorithm was presented in terms of the position and velocity Root Mean Squared Error (RMSE). The POSE algorithm achieves a much smaller RMSE then the always monitoring network, random scheduling network, and trigger-based activation network. This is because the POSE network incorporates distributed fusion into the DSN, which minimizes estimate error. Therefore, the POSE algorithm achieves low missed detection rates, significant energy savings, and accurate tracking accuracy.

Although this theme extends the network lifetime, the POSE algorithms design enables redundant nodes in a high power consuming state around the target's location. Additionally, in practical applications, not all of the targets within the deployment region are of interest to the mission operator. Therefore, if the network is enabling redundant sensor nodes around every target that travels through the deployment region, significant energy is being wasted. Therefore, the second theme of this thesis extends the POSE algorithm to address the above challenges.

6.2 POSE using Distributed Classification, Clustering, and Control (POSE.3C)

To address the above issues, the *Prediction-based Opportunistic Sensing using Distributed Classification, Clustering and Control* (POSE.3C) algorithm was developed as the second theme of this thesis to further improve the energy-efficiency of the DSN. The POSE.3C algorithm continues using the distributed supervisor (i.e. PFSA) developed in the POSE algorithm, except the state transition probabilities are updated based on the target's class as well as the target's predicted location. This new algorithm is composed of two additional steps in the control scheme, 1) Target classification and 2) Sensor selection.

The sensor nodes that are tracking the target perform target classification to identify if the target is a *Target Of Interest* (TOI) or a *Target Not Of Interest* (TNOI). For example, in a boarder surveillance application, the TOIs could be humans and vehicles, while the TNOIs could be animals. Then, the sensor nodes that use received information to update the PFSA will perform distributed fusion to identify the type of target present. Next, a distributed clustering approach is employed for sensor selection to select $N_{sel} > 1$ sensor nodes to track TOI to improve estimation, while only selecting $N_{sel} = 1$ sensor nodes to track TNOI to maintain awareness. In this regard, the DSN opportunistically senses TOI and conserves energy around TNOI.

Therefore, the novel contributions of the second theme are as follows:

- Development of a distributed algorithm that facilitates 3C network autonomy with the following attributes:
  (i) distributed classification and utilization of the target class information to govern the cluster size for tracking,
  (ii) distributed clustering via a) fusing the target state estimates and predicting the next state, and b) sensor node selection around the target's predicted state, and
  (iii) distributed supervisory control to enable/disable multi-modal sensing devices for energy-efficiency.

- Theoretical analysis of the network's performance characteristics: (i) Expected energy consumption, (ii) Expected network lifetime, (iii) Probability of missed detection for a target birth and for a mature target.

The first major result of the POSE.3C algorithm is the Theoretical properties of the network. This theme developed 3 theorems that allow the network designer to predict the average energy consumption, network lifetime, and missed detection characteristics based on the multi-modal sensor node design, network density, and PFSA design parameters. The theoretical properties were validated thoroughly via simulations.

Additionally, the POSE.3C algorithm was validated by comparing the network lifetime and tracking errors against the Autonomous Node Selection (ANS) algorithm [29], a Random Scheduling network, Trigger-based activation network. The network lifetime result showed that the POSE.3C network is able to achieve the longest network lifetime if only TNOI are traveling through the network. This result shows that by incorporating classification feedback into the control loop, the network lifetime can be drastically increased when TNOI are traveling through the network. Additionally, the POSE.3C network was able to achieve a longer network lifetime as compared to the other method even when only TOI were traveling through the network. This result shows that using multiple operating conditions and distributed clustering, the number of active sensors around the target can be minimized, allowing the sensor nodes to intelligently perform their tasks while conserving energy when not needed.

Furthermore, the tracking accuracy results show that the POSE.3C network is able to achieve similar performance as the ANS algorithm. The ANS algorithm selects sensors that minimize the measurement covariance error, while the POSE.3C algorithm minimizes error while incorporating uniform energy depletion. Therefore, at the cost of uniform energy depletion, the POSE.3C algorithm has a slightly larger RMSE, however, the network lifetime is significantly longer for the POSE.3C network.

The POSE and POSE.3C network allow for an extended network lifetime through energy-efficient control. However, if multiple co-located sensor nodes fail, a coverage gap forms and the network is not designed to ensure target coverage within the gap. Therefore, the POSE.3C algorithm was extended to incorporate an approach that learns and reacts to the density of the network around the target's predicted position in order to ensure resilient target coverage.

6.3 POSE for Resilience (POSE.R)

The third theme of this thesis further extends the POSE and POSE.3C algorithm by incorporating resilient target coverage into the DSN by developing the *Prediction-based Opportunistic Sensing for Resilience* (POSE.R) algorithm. Here, the multi-modal sensor nodes are assumed to contain a high power sensing device that can vary it's sensing range at the cost of consuming more energy. The POSE.R algorithm incorporates an adaptive sensor selection approach that selects the optimal sensor nodes and their sensing ranges to track the target and ensure target coverage in the presence of failures and non-uniform node locations. The sensor selection approach adapts to the density of active sensor nodes around the target's predicted location. For high density regions with $\geq N_{sel}$ nodes, an *Energy-based Geometric Dilution of Precision* (EGDOP) sensor selection method is employed to select and activate geometrically diverse nodes with high remaining energy to cover the target with their minimum sensing range.

When a target is predicted to travel through a low density region, the sensor nodes may need to expand their sensing range to accommodate for a coverage gap or an insufficient number of nodes. To identify the best sensing ranges to cover the target while minimizing energy consumption, a Game Theoretic sensor range selection method is employed using *Potential Games*. The objective function of the game selects N_{sel} nodes and their sensing ranges to cover the target's prediction location while minimizing energy wastage by reducing redundant coverage. In this regard, the DSN becomes resilient to coverage gaps by opportunistically covering the gaps when a target is predicted to travel through it, while operating in an energy-efficient manner when the target travels through high density regions.

Therefore, the novel contributions of the POSE.R algorithm is the development of a distributed algorithm that facilitates resilient target coverage and tracking with the following attributes:

- distributed coverage gap identification method that does not rely on active or passive monitoring methods,
- distributed node selection approach that adapts to the network density around a target's predicted location via a) EGDOP selection criteria for high density regions, and b) a Game

Theoretic node and range selection method using potential games for low density regions.

- distributed supervisory control strategy to enable/disable multi-modal sensing devices for energy-efficiency.

The POSE.R algorithm was simulated and compared against the ANS algorithm, Random Scheduling, and a Trigger-based Activation network. The methods available in literature do not consider adjustable sensing ranges. Therefore, the other methods were simulated for each of the adjustable ranges of the POSE.R algorithm to show the benefits of allowing for range adaptation.

The first result compared the probability of missed detection against the other method. This showed that for low network densities, the other networks required a very large sensor range in order to achieve similar missed detection characteristics as the POSE.R network. Additionally for low sensing ranges, the other networks would require a very large network density to achieve the same missed detections as the POSE.R network. Therefore, to achieve the same performance as the POSE.R network, a combination of large sensing ranges or high network density must be used to achieve the same performance.

In addition to achieving low missed detection characteristics, the POSE.R algorithm achieves high tracking accuracy. In this theme, the sensor selection cost function is very similar to the ANS cost function which allows the tracking accuracy to be the same as ANS and significantly lower than the other methods.

Furthermore, the energy consumption characteristics showed that the POSE.R algorithm can significantly save energy as compared to the other methods. However, if the network density is low and the sensing range for the other networks is low, the energy consumption of the POSE.R algorithm around the target is slightly larger. This is because the POSE.R algorithm is adjusting the sensing ranges at the cost of energy to achieve low missed detection rates and increased target coverage. However, as the network density and sensing range increase, the POSE.R algorithm provides significant energy savings by reducing the sensing range if the target travels through a region of high density sensor nodes.

Finally, the resilience of the POSE.R network was presented by simulating a target traveling through a coverage gap. The gaps size was varied and the results show that the POSE.R algorithm is able to achieve high detection probabilities as compared to the other methods, even when the target is in the center of the gap. However, when the other methods have a very large sensing range, the detection performance is the same as the POSE.R network. Although the detection performance is the same, the energy consumption of the POSE.R network is lower then the other methods, thus shows that the POSE.R network is energy-efficient.

These results show that incorporating adjustable ranges into the DSN provides resilient target coverage to coverage gaps and non-uniformly distributed sensor placement. In addition, the network is able to achieve an extended network lifetime, low missed detection rates, and accurate state estimation.

BIBLIOGRAPHY

[1] Vijay Gupta, Timothy H Chung, Babak Hassibi, and Richard M Murray. “On a stochastic sensor selection algorithm with applications in sensor scheduling and sensor coverage”. In: *Automatica* 42.2 (2006), pp. 251–260.

[2] Kushal Mukherjee, Shalabh Gupta, Asok Ray, and Thomas A Wettergren. “Statistical-mechanics-inspired optimization of sensor field configuration for detection of mobile targets”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 41.3 (2011), pp. 783–791.

[3] Junfeng Wu, Qing-Shan Jia, Karl Henrik Johansson, and Ling Shi. “Event-based sensor data scheduling: Trade-off between communication rate and estimation quality”. In: *IEEE Transactions on Automatic Control* 58.4 (2013), pp. 1041–1046.

[4] Shanying Zhu, Cailian Chen, Wenshuang Li, Bo Yang, and Xinping Guan. “Distributed optimal consensus filter for target tracking in heterogeneous sensor networks”. In: *IEEE Transactions on Cybernetics* 43.6 (2013), pp. 1963–1976.

[5] Hamid Mahboubi, Walid Masoudimansour, Amir G Aghdam, and Kamran Sayrafian-Pour. “An Energy-Efficient Target-Tracking Strategy for Mobile Sensor Networks”. In: *IEEE Transactions on Cybernetics* PP.99 (2016), pp. 1–13.

[6] Qiang Zhang, Meiqin Liu, and Senlin Zhang. “Node topology effect on target tracking based on UWSNs using quantized measurements”. In: *IEEE Transactions on Cybernetics* 45.10 (2015), pp. 2323–2335.

[7] Xin Jin, Soumalya Sarkar, Asok Ray, Shalabh Gupta, and Thyagaraju Damarla. “Target detection and classification using seismic and PIR sensors”. In: *IEEE Sensors Journal* 12.6 (2012), pp. 1709–1718.

[8] **James Zachary Hare**, Shalabh Gupta, and James Wilson. “Decentralized smart sensor scheduling for multiple target tracking for border surveillance”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2015, pp. 3265–3270.

[9] Antonio-Javier Garcia-Sanchez, Felipe Garcia-Sanchez, Fernando Losilla, Pawel Kulakowski, Joan Garcia-Haro, Alejandro Rodriguez, Jose-Vicente Lopez-Bao, and Francisco Palomares. “Wireless sensor network deployment for monitoring wildlife passages”. In: *Sensors* 10.8 (2010), pp. 7236–7262.

[10] Robert Szewczyk, Eric Osterweil, Joseph Polastre, Michael Hamilton, Alan Mainwaring, and Deborah Estrin. “Habitat monitoring with sensor networks”. In: *Communications of the ACM* 47.6 (2004), pp. 34–40.

[11] Geoffrey Werner-Allen, Konrad Lorincz, Mario Ruiz, Omar Marcillo, Jeff Johnson, Jonathan Lees, and Matt Welsh. “Deploying a wireless sensor network on an active volcano”. In: *IEEE Internet Computing* 10.2 (2006), pp. 18–25.

[12] M Scaioni, P Lu, T Feng, W Chen, G Qiao, H Wu, X Tong, W Wang, and R Li. “Analysis of spatial sensor network observations during landslide simulation experiments”. In: *European Journal of Environmental and Civil Engineering* 17.9 (2013), pp. 802–825.

[13] Alexandros Pantelopoulos and Nikolaos G Bourbakis. “A survey on wearable sensor-based systems for health monitoring and prognosis”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 40.1 (2010), pp. 1–12.

[14] Hyun Lee, Jae Sung Choi, and Ramez Elmasri. “A static evidential network for context reasoning in home-based care”. In: *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 40.6 (2010), pp. 1232–1243.

[15] T Semertzidis, K Dimitropoulos, A Koutsia, and N Grammalidis. “Video sensor network for real-time traffic monitoring and surveillance”. In: *IET Intelligent Transport Systems* 4.2 (2010), pp. 103–112.

[16] Ali EIKateeb. “Mote design supported with remote hardware modifications capability for wireless sensor network applications”. In: *Int. J. Adv. Smart Sens. Netw. Syst* 3 (2013), pp. 13–21.

[17] Giuseppe Anastasi, Marco Conti, Mario Di Francesco, and Andrea Passarella. “Energy conservation in wireless sensor networks: A survey”. In: *Ad hoc networks* 7.3 (2009), pp. 537–568.

[18] Anish Arora, Prabal Dutta, Sandip Bapat, Vinod Kulathumani, Hongwei Zhang, Vinayak Naik, Vineet Mittal, Hui Cao, Murat Demirbas, Mohamed Gouda, et al. “A line in the sand: a wireless sensor network for target detection, classification, and tracking”. In: *Computer Networks* 46.5 (2004), pp. 605–634.

[19] Yun Li, Lucas W Krakow, Edwin KP Chong, and Kenneth N Groom. “Approximate stochastic dynamic programming for sensor scheduling to track multiple targets”. In: *Digital Signal Processing* 19.6 (2009), pp. 978–989.

[20] Artemis C Voulkidis and Panayotis G Cottis. “Optimal Node Allocation in Multiservice WSNs Based on Correlated Strategy”. In: *IEEE Transactions on Wireless Communications* 15.6 (2016), pp. 4196–4205.

[21] Ameer Ahmed Abbasi and Mohamed Younis. “A survey on clustering algorithms for wireless sensor networks”. In: *Computer communications* 30.14 (2007), pp. 2826–2841.

[22] Wei-Peng Chen, Jennifer C Hou, and Lui Sha. “Dynamic clustering for acoustic target tracking in wireless sensor networks”. In: *IEEE Transactions on Mobile Computing* 3.3 (2004), pp. 258–271.

[23] Ossama Younis and Sonia Fahmy. “HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks”. In: *IEEE Transactions on Mobile Computing* 3.4 (2004), pp. 366–379.

[24] Erdal Cayirci, Hakan Tezcan, Yasar Dogan, and Vedat Coskun. “Wireless sensor networks for underwater survelliance systems”. In: *Ad Hoc Networks* 4.4 (2006), pp. 431–446.

[25] Sundeep Pattem, Sameera Poduri, and Bhaskar Krishnamachari. “Energy-quality tradeoffs for target tracking in wireless sensor networks”. In: *Information processing in sensor networks*. Springer. 2003, pp. 32–46.

[26] Jiming Chen, Kejie Cao, Keyong Li, and Youxian Sun. “Distributed sensor activation algorithm for target tracking with binary sensor networks”. In: *Cluster Computing* 14.1 (2011), pp. 55–64.

[27] Wei Li and Wei Zhang. “Sensor selection for improving accuracy of target localisation in wireless visual sensor networks”. In: *IET wireless sensor systems* 2.4 (2012), pp. 293–301.

[28] Lance M Kaplan. “Global node selection for localization in a distributed sensor network”. In: *IEEE Transactions on Aerospace and Electronic systems* 42.1 (2006), pp. 113–135.

[29] Lance M Kaplan. “Local node selection for localization in a distributed sensor network”. In: *IEEE Transactions on Aerospace and Electronic Systems* 42.1 (2006), pp. 136–146.

[30] Ying-Chih Chen and Chih-Yu Wen. “Decentralized Cooperative TOA/AOA Target Tracking for Hierarchical Wireless Sensor Networks”. In: *Sensors* 12.11 (2012), pp. 15308–15337.

[31] Dragana Bajovic, Bruno Sinopoli, and João Xavier. “Sensor selection for event detection in wireless sensor networks”. In: *IEEE Transactions on Signal Processing* 59.10 (2011), pp. 4938–4953.

[32] Tom Vercauteren, Dong Guo, and Xiaodong Wang. “Joint multiple target tracking and classification in collaborative sensor networks”. In: *IEEE Journal on selected areas in communications* 23.4 (2005), pp. 714–723.

[33] Mohamed Younis, Izzet F Senturk, Kemal Akkaya, Sookyoung Lee, and Fatih Senel. “Topology management techniques for tolerating node failures in wireless sensor networks: A survey”. In: *Computer Networks* 58 (2014), pp. 254–283.

[34] Raymond Mulligan and Habib M Ammari. “Coverage in wireless sensor networks: A survey”. In: *network protocols and algorithms* 2.2 (2010), pp. 27–53.

[35] Mengjie Yu, Hala Mokhtar, and Madjid Merabti. “Fault management in wireless sensor networks”. In: *IEEE Wireless Communications* 14.6 (2007).

[36] Lilia Paradis and Qi Han. “A survey of fault management in wireless sensor networks”. In: *Journal of Network and systems management* 15.2 (2007), pp. 171–190.

[37] H. M. Ammari and S. Das. “A Study of k-Coverage and Measures of Connectivity in 3D Wireless Sensor Networks”. In: *IEEE Transactions on Computers* 59.2 (2010), pp. 243–257.

[38] Sookyoung Lee, Mohamed Younis, and Meejeong Lee. “Connectivity restoration in a partitioned wireless sensor network with assured fault tolerance”. In: *Ad Hoc Networks* 24 (2015), pp. 1–19.

[39] Stefano Chessa and Piero Maestrini. “Fault recovery mechanism in single-hop sensor networks”. In: *Computer communications* 28.17 (2005), pp. 1877–1886.

[40] Qinghua Li and Guohong Cao. “Mitigating routing misbehavior in disruption tolerant networks”. In: *IEEE transactions on information forensics and security* 7.2 (2012), pp. 664–675.

[41] **James Zachary Hare**, Shalabh Gupta, and Thomas Wettergren. “Prediction-based Opportunistic Sensing for Energy-efficient Sensor Networks using Distributed Supervisors”. In: *IEEE Transactions on Cybernetics* (2017). DOI: `10.1109/TCYB.2017.2727981`.

[42] **James Zachary Hare**, Shalabh Gupta, and Thomas Wettergren. “POSE.3C: Prediction-based Opportunistic Sensing using Distributed Classification, Clustering and Control - A Network-centric Autonomy Concept”. In: *IEEE Transactions on Control of Network Systems, **Under Review*** (2017).

[43] **James Zachary Hare**, Shalabh Gupta, Junnan Song, and Thomas Wettergren. “POSE.R: Prediction-based Opportunistic Sensing for Resilient and Efficient Sensor Networks”. In: *ACM Transactions on Sensor Networks, **Ready for submission*** (2018).

[44] Mingming Lu, Jie Wu, Mihaela Cardei, and Minglu Li. “Energy-efficient connected coverage of discrete targets in wireless sensor networks”. In: *Networking and mobile computing* (2005), pp. 43–52.

[45] Yaakov Bar-Shalom, X Rong Li, and Thiagalingam Kirubarajan. *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

[46] Yi Zou and Krishnendu Chakrabarty. “Sensor deployment and target localization in distributed sensor networks”. In: *ACM Transactions on Embedded Computing Systems (TECS)* 3.1 (2004), pp. 61–91.

[47] Thomas A Wettergren. “Performance of search via track-before-detect for distributed sensor networks”. In: *IEEE Transactions on Aerospace and Electronic systems* 44.1 (2008), pp. 314–325.

[48] Yaakov Bar-Shalom, Fred Daum, and Jim Huang. “The probabilistic data association filter”. In: *IEEE Control Systems* 29.6 (2009), pp. 82–100.

[49] Chong Liu, Kui Wu, Yang Xiao, and Bo Sun. “Random coverage with guaranteed connectivity: joint scheduling for wireless sensor networks”. In: *IEEE Transactions on parallel and distributed systems* 17.6 (2006), pp. 562–575.

[50] Jianyong Lin, Wendong Xiao, Frank L Lewis, and Lihua Xie. “Energy-efficient distributed adaptive multisensor scheduling for target tracking in wireless sensor networks”. In: *IEEE Transactions on Instrumentation and Measurement* 58.6 (2009), pp. 1886–1896.

[51] Aman Kansal, Jason Hsu, Sadaf Zahedi, and Mani B Srivastava. “Power management in energy harvesting sensor networks”. In: *ACM Transactions on Embedded Computing Systems (TECS)* 6.4 (2007), p. 32.

[52] M Chamara Ranasingha, Manohar N Murthi, Kamal Premaratne, and Xingzhe Fan. “Transmission rate allocation in multisensor target tracking over a shared network”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 39.2 (2009), pp. 348–362.

[53] Juan Liu, Maurice Chu, and James E Reich. “Multitarget tracking in distributed sensor networks”. In: *IEEE Signal Processing Magazine* 24.3 (2007), pp. 36–46.

[54] Xuanwen Luo, Ming Dong, and Yinlun Huang. “On distributed fault-tolerant detection in wireless sensor networks”. In: *IEEE Transactions on Computers* 55.1 (2006), pp. 58–70.

[55] **James Zachary Hare**, Shalabh Gupta, and Junnan Song. “Distributed smart sensor scheduling for underwater target tracking”. In: *Oceans’14 MTS/IEEE*. 2014.

[56] Xuxun Liu. “A Deployment Strategy for Multiple Types of Requirements in Wireless Sensor Networks”. In: *IEEE Transactions on Cybernetics* 45.10 (Oct. 2015), pp. 2364–2376.

[57] Chia-Pang Chen, Subhas Chandra Mukhopadhyay, Cheng-Long Chuang, Tzu-Shiang Lin, Min-Sheng Liao, Yung-Chung Wang, and Joe-Air Jiang. “A Hybrid Memetic Framework for Coverage Optimization in Wireless Sensor Networks”. In: *IEEE Transactions on Cybernetics* 45.10 (Oct. 2015), pp. 2309–2322.

[58] Krishnendu Chakrabarty, S Sitharama Iyengar, Hairong Qi, and Eungchun Cho. “Grid coverage for surveillance and target location in distributed sensor networks”. In: *IEEE Transactions on Computers* 51.12 (2002), pp. 1448–1453.

[59] Yuan Song, Bing Wang, Zhijie Shi, Krishna R Pattipati, and Shalabh Gupta. “Distributed algorithms for energy-efficient even self-deployment in mobile sensor networks”. In: *IEEE Transactions on Mobile Computing* 13.5 (2014), pp. 1035–1047.

[60] Wendong Xiao, Sen Zhang, Jianyong Lin, and Chen Khong Tham. “Energy-efficient adaptive sensor scheduling for target tracking in wireless sensor networks”. In: *Journal of Control Theory and Applications* 8.1 (2010), pp. 86–92.

[61] Yonggui Liu, Bugong Xu, and Linfang Feng. “Energy-balanced multiple-sensor collaborative scheduling for maneuvering target tracking in wireless sensor networks”. In: *Journal of Control Theory and Applications* 9.1 (2011), pp. 58–65.

[62] Engin Masazade, Makan Fardad, and Pramod K Varshney. “Sparsity-promoting extended Kalman filtering for target tracking in wireless sensor networks”. In: *IEEE Signal Processing Letters* 19.12 (2012), pp. 845–848.

[63] Xiaojing Shen and Pramod K Varshney. “Sensor selection based on generalized information gain for target tracking in large sensor networks”. In: *IEEE Transactions on Signal Processing* 62.2 (2014), pp. 363–375.

[64] Shuchin Aeron, Venkatesh Saligrama, and David A Castanon. “Efficient sensor management policies for distributed target tracking in multihop sensor networks”. In: *IEEE Transactions on Signal Processing* 56.6 (2008), pp. 2562–2574.

[65] Ossama Younis, Marwan Krunz, and Srinivasan Ramasubramanian. “Node clustering in wireless sensor networks: recent developments and deployment challenges”. In: *IEEE Network* 20.3 (2006), pp. 20–25.

[66] Jianghai Li, Qing-Shan Jia, Xiaohong Guan, and Xi Chen. “Tracking a moving object via a sensor network with a partial information broadcasting scheme”. In: *Information Sciences* 181.20 (2011), pp. 4733–4753.

[67] Chuan Feng, Lizhi Yang, and Jerzy W Rozenblit. “Adaptive Tracking in Energy Sensitive Distributed Wireless Sensor Networks.” In: *Ad Hoc & Sensor Wireless Networks* 12.1-2 (2011), pp. 55–77.

[68] Enrique Vidal, Franck Thollard, Colin De La Higuera, Francisco Casacuberta, and Rafael C Carrasco. “Probabilistic finite-state machines-part I”. In: *IEEE transactions on pattern analysis and machine intelligence* 27.7 (2005), pp. 1013–1025.

[69] Y. Bar-Shalom. “On the track-to-track correlation problem”. In: *IEEE Transactions on Automatic Control* 26.2 (Apr. 1981), pp. 571–572.

[70] Yaakov Bar-Shalom, Peter K Willett, and Xin Tian. *Tracking and data fusion: A Handbook of Algorithms*. YBS Publishing, 2011.

[71] Y Bar-Shalom, KC Chang, and HAP Blom. “Automatic track formation in clutter with a recursive algorithm”. In: *Proceedings of the IEEE Conference on Decision and Control*. IEEE. 1989, pp. 1402–1408.

[72] Stefano Coraluppi and Craig Carthel. “Distributed tracking in multistatic sonar”. In: *IEEE Transactions on Aerospace and Electronic Systems* 41.3 (2005), pp. 1138–1147.

[73] Marco F Duarte and Yu Hen Hu. “Vehicle classification in distributed sensor networks”. In: *Journal of Parallel and Distributed Computing* 64.7 (2004), pp. 826–838.

[74] Norman Dziengel, Martin Seiffert, Marco Ziegert, Stephan Adler, Stefan Pfeiffer, and Jochen Schiller. “Deployment and evaluation of a fully applicable distributed event detection system in Wireless Sensor Networks”. In: *Ad Hoc Networks* 37 (2016), pp. 160–182.

[75] Kushal Mukherjee, Shalabh Gupta, Asok Ray, and Shashi Phoha. “Symbolic Analysis of Sonar Data for Underwater Target Detection”. In: *IEEE Journal of Oceanic Engineering* 36.2 (2011), pp. 219–230.

[76] Donka Angelova and Lyudmila Mihaylova. “Joint target tracking and classification with particle filtering and mixture Kalman filtering using kinematic radar information”. In: *Digital Signal Processing* 16.2 (2006), pp. 180–204.

[77] Lance M Kaplan. “Global node selection for localization in a distributed sensor network”. In: *IEEE Transactions on Aerospace and Electronic systems* 42.1 (2006), pp. 113–135.

[78] Emre Ertin, John W Fisher, and Lee C Potter. “Maximum mutual information principle for dynamic sensor query problems”. In: *Information processing in sensor networks*. Springer. 2003, pp. 405–416.

[79] Mohammad Zoghi and Mohammad Hosein Kahaei. “Adaptive sensor selection in wireless sensor networks for target tracking”. In: *IET Signal Processing* 4.5 (2010), pp. 530–536.

[80] Farhad Ghassemi and Vikram Krishnamurthy. “Decentralized node selection for localization in wireless unattended ground sensor networks”. In: *International Conference on Sensor Technologies and Applications, SENSORCOMM*. IEEE. 2008, pp. 294–299.

[81] “HTI Hydrophone `http://www.hightechincusa.com/products/hydrophones/`”. In: ().

[82] “M3 Sonar `https://www.km.kongsberg.com`”. In: ().

[83] Chi-Fu Huang and Yu-Chee Tseng. “The coverage problem in a wireless sensor network”. In: *Mobile Networks and Applications* 10.4 (2005), pp. 519–528.

[84] James Zachary Hare, Shalabh Gupta, and Thomas Wettergren. “POSE: Prediction-Based Opportunistic Sensing for Energy Efficiency in Sensor Networks Using Distributed Supervisors”. In: *IEEE Transactions on Cybernetics* (2017).

[85] Mihaela Cardei, Jie Wu, Mingming Lu, and Mohammad O Pervaiz. “Maximum network lifetime in wireless sensor networks with adjustable sensing ranges”. In: *Wireless and Mobile Computing, Networking and Communications, 2005.(WiMob’2005), IEEE International Conference on*. Vol. 3. IEEE. 2005, pp. 438–445.

[86] Bhaskar Krishnamachari and Sitharama Iyengar. “Distributed Bayesian algorithms for fault-tolerant event region detection in wireless sensor networks”. In: *IEEE Transactions on Computers* 53.3 (2004), pp. 241–250.

[87] Min Ding, Fang Liu, Andrew Thaeler, Dechang Chen, and Xiuzhen Cheng. “Fault-tolerant target localization in sensor networks”. In: *EURASIP Journal on Wireless Communications and Networking* 2007.1 (2007), pp. 19–19.

[88] Prasanna Sridhar, Asad M Madni, and Mo Jamshidi. “Hierarchical aggregation and intelligent monitoring and control in fault-tolerant wireless sensor networks”. In: *IEEE Systems Journal* 1.1 (2007), pp. 38–54.

[89] Arslan Munir and Ann Gordon-Ross. “Markov modeling of fault-tolerant wireless sensor networks”. In: *Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on*. IEEE. 2011, pp. 1–6.

[90] Richard R Brooks, Parameswaran Ramanathan, and Akbar M Sayeed. “Distributed target classification and tracking in sensor networks”. In: *Proceedings of the IEEE* 91.8 (2003), pp. 1163–1171.

[91] Nadeem Ahmed, Salil S Kanhere, and Sanjay Jha. “The holes problem in wireless sensor networks: a survey”. In: *ACM SIGMOBILE Mobile Computing and Communications Review* 9.2 (2005), pp. 4–18.

[92] Ning Li and Jennifer C Hou. “Localized fault-tolerant topology control in wireless ad hoc networks”. In: *IEEE Transactions on Parallel and Distributed Systems* 17.4 (2006), pp. 307–320.

[93] Xiaofeng Han, Xiang Cao, Errol L Lloyd, and Chien-Chung Shen. “Fault-tolerant relay node placement in heterogeneous wireless sensor networks”. In: *IEEE Transactions on Mobile Computing* 9.5 (2010), pp. 643–656.

[94] Habib Ammari and Sajal Das. “Coverage, connectivity, and fault tolerance measures of wireless sensor networks”. In: *Stabilization, Safety, and Security of Distributed Systems* (2006), pp. 35–49.

[95] Wolfgang W Bein, Doina Bein, and Srilaxmi Malladi. “Reliability and fault tolerance of coverage models for sensor networks”. In: *International Journal of Sensor Networks* 5.4 (2009), pp. 199–209.

[96] Gaurav Gupta and Mohamed Younis. “Fault-tolerant clustering of wireless sensor networks”. In: *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*. Vol. 3. IEEE. 2003, pp. 1579–1584.

[97] Sania Bhatti, Jie Xu, and Mohsin Memon. “Clustering and fault tolerance for target tracking using wireless sensor networks”. In: *IET wireless sensor systems* 1.2 (2011), pp. 66–73.

[98] Md Azharuddin, Pratyay Kuila, and Prasanta K Jana. “Energy efficient fault tolerant clustering and routing algorithms for wireless sensor networks”. In: *Computers & Electrical Engineering* 41 (2015), pp. 177–190.

[99] Mihaela Cardei, Jie Wu, and Mingming Lu. “Improving network lifetime using sensors with adjustable sensing ranges”. In: *International Journal of Sensor Networks* 1.1-2 (2006), pp. 41–49.

[100] Mingming Lu, Jie Wu, Mihaela Cardei, and Minglu Li. “Energy-efficient connected coverage of discrete targets in wireless sensor networks”. In: *International Journal of Ad Hoc and Ubiquitous Computing* 4.3-4 (2009), pp. 137–147.

[101] Akshaye Dhawan, Chinh T Vu, Alexander Zelikovsky, Yingshu Li, and Sushil K Prasad. “Maximum lifetime of sensor networks with adjustable sensing range”. In: *Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2006. SNPD 2006. Seventh ACIS International Conference on*. IEEE. 2006, pp. 285–289.

[102] Raffaele Cerulli, R De Donato, and Andrea Raiconi. “Exact and heuristic methods to maximize network lifetime in wireless sensor networks with adjustable sensing ranges”. In: *European Journal of Operational Research* 220.1 (2012), pp. 58–66.

[103] Hosein Mohamadi, Shaharuddin Salleh, and Mohd Norsyarizad Razali. “Heuristic methods to maximize network lifetime in directional sensor networks with adjustable sensing ranges”. In: *Journal of Network and Computer Applications* 46 (2014), pp. 26–35.

[104] Zongheng Zhou, Samir R Das, and Himanshu Gupta. “Variable radii connected sensor cover in sensor networks”. In: *ACM Transactions on Sensor Networks (TOSN)* 5.1 (2009), p. 8.

[105] Novella Bartolini, Tiziana Calamoneri, Tom La Porta, Chiara Petrioli, and Simone Silvestri. “Sensor activation and radius adaptation (SARA) in heterogeneous sensor networks”. In: *ACM Transactions on Sensor Networks (TOSN)* 8.3 (2012), p. 24.

[106] Hosein Mohamadi, Shaharuddin Salleh, Mohd Norsyarizad Razali, and Sara Marouf. “A new learning automata-based approach for maximizing network lifetime in wireless sensor networks with adjustable sensing ranges”. In: *Neurocomputing* 153 (2015), pp. 11–19.

[107] Akshaye Dhawan, Aung Aung, and Sushil K Prasad. “Distributed Scheduling of a Network of Adjustable Range Sensors for Coverage Problems.” In: *ICISTM*. Springer. 2010, pp. 123–132.

[108] Roger Myerson. *Game theory*. Harvard university press, 2013.

[109] Yang Song, Starsky Wong, and Kang-Won Lee. “Optimal gateway selection in multi-domain wireless networks: a potential game perspective”. In: *Proceedings of the 17th annual international conference on Mobile computing and networking*. ACM. 2011, pp. 325–336.

[110] Haibo Dai, Yongming Huang, and Luxi Yang. “Game theoretic max-logit learning approaches for joint base station selection and resource allocation in heterogeneous networks”. In: *IEEE Journal on Selected Areas in Communications* 33.6 (2015), pp. 1068–1081.

APPENDIX A

AUTOMATIC TRACK FORMATION (ATF)

In this work, a track is assumed to be initialized by the ATF method. This approach assumes that the target present in the deployment region can travel at a maximum velocity V_{max}. Therefore, if two consecutive measurements, $\mathbf{z}_j(k-1)$ at time $k-1$ and $\mathbf{z}_j(k)$ time k are not associated to a previous state estimate, then the following condition is checked. If

$$|\mathbf{z}_j(k) - \mathbf{z}_j(k-1)| \leq V_{max}\Delta T, \tag{A.1}$$

then a new track is formed as follows.

$$\begin{aligned} \hat{\mathbf{x}}^{s_i,c}(k|k) &= \Big[z_j^x(k), \frac{z_j^x(k) - z_j^x(k-1)}{\Delta T}, \\ &\quad z_j^y(k), \frac{z_j^y(k) - z_j^y(k-1)}{\Delta T}\Big]' \end{aligned} \tag{A.2}$$

$$\hat{\mathbf{\Sigma}}^{s_i}(k|k) = \begin{bmatrix} \sigma_x^2 & \frac{\sigma_x^2}{\Delta T} & 0 & 0 \\ \frac{\sigma_x^2}{\Delta T} & \frac{2\sigma_x^2}{\Delta T^2} & 0 & 0 \\ 0 & 0 & \sigma_y^2 & \frac{\sigma_y^2}{\Delta T} \\ 0 & 0 & \frac{\sigma_y^2}{\Delta T} & \frac{2\sigma_y^2}{\Delta T^2} \end{bmatrix} \tag{A.3}$$

where σ_x and σ_y are the Cartesian measurement noises converted from the polar domain.

APPENDIX B

JOINT DATA ASSOCIATION FILTER (JPDA) FILTER

The JPDA filter is a state estimation method used to track multiple targets in the presence of clutter. To perform state estimation locally on each sensor, sensor node s_i first receives a set of measurements, $\mathbf{z}(k)$, and then updates the previous target states as follows.

B.0.1 State Prediction

First, sensor node s_i computes the expected state vector, measurement, and state covariance matrices during the next time step as follows.

$$\begin{aligned}
\hat{\mathbf{x}}^{s_i,c}(k|k-1) &= \mathbf{f}(\hat{\mathbf{x}}^{s_i,c}(k-1|k-1), k-1) \\
\hat{\mathbf{z}}^{s_i,c}(k|k-1) &= \mathbf{h}(\hat{\mathbf{x}}^{s_i,c}(k|k-1), k) \\
\hat{\mathbf{\Sigma}}^{s_i,c}(k|k-1) &= \mathbf{F}(k-1)\hat{\mathbf{\Sigma}}^{s_i,c}(k-1|k-1)\mathbf{F}(k-1)' + \mathbf{Q}
\end{aligned} \tag{B.1}$$

where $\hat{\mathbf{x}}^{s_i,c}(k-1|k-1)$ is the target state, $\hat{\mathbf{\Sigma}}^{s_i,c}(k-1|k-1)$ is the target state covariance, $\mathbf{f}(\hat{\mathbf{x}}^{s_i,c}(k-1|k-1), k-1)$ is the state transition matrix, $\mathbf{h}(\hat{\mathbf{x}}^{s_i,c}(k|k-1), k)$ is the measurement model, and $\mathbf{F}(k-1)$ is the Jacobian of the state transition matrix.

B.0.2 Measurement Validation

Next, the set of validated measurements that fall within the validation gate of state estimate c is computed as follows.

$$\bar{\mathbf{z}}^{s_i,c}(k) = \{\mathbf{z}_j(k) \in \mathbf{z}(k) : \mathbf{v}^{s_i,c}(k)' \mathbf{S}^{s_i,c}(k)^{-1} \mathbf{v}_j^{s_i,c}(k) \leq \eta\} \tag{B.2}$$

where $\mathbf{v}_j^{s_i,c}(k) = [\mathbf{z}_j(k) - \hat{\mathbf{z}}^{s_i,c}(k|k-1)]$ is the innovation, η is the validation gate threshold,

$$\mathbf{S}^{s_i,c}(k) = \mathbf{H}(k)\hat{\boldsymbol{\Sigma}}^{s_i,c}(k|k-1)\mathbf{H}(k)' + \mathbf{R}(k) \tag{B.3}$$

is the innovation covariance, $\mathbf{H}(k)$ is the Jacobian of the measurement model, and $\mathbf{R}(k)$ is the covariance of the measurement noise.

B.0.3 Joint Association Probabilities

After measurement validation, the joint association probabilities of the event, $A_{j,c}(k)$, that measurement j at time k originated from target c as follows.

$$P(A(k)|\mathbf{z}(k)) = \frac{1}{n_1} \prod_j \{\lambda^{-1}\Lambda_{j,c}\}^{\varsigma_j} \prod_c \{P_D^{s_i,c}\}^{\delta_c}\{1 - P_D^{s_i,c}\}^{1-\delta_c} \tag{B.4}$$

where n_1 is a normalization constant, λ is spatial density of false measurements, $\varsigma_j = 1$ if $\mathbf{z}_j(k) \in \bar{\mathbf{z}}^{s_i,c}(k)$ and $\varsigma_j = 0$ if not, δ_c is the target detection indicator, $P_D^{s_i,c}$ is the probability of detecting target c, and

$$\Lambda_{j,c} = \mathcal{N}\left[\mathbf{z}_j(k); \hat{\mathbf{z}}^{s_i,c}(k|k-1), \mathbf{S}^{s_i,c}(k)\right]. \tag{B.5}$$

B.0.4 State Estimation

Then, assuming that the target states conditions on the previous measurements are mutually independent, the marginal association probabilities, $P_{jc}(k)$, can be computed.

$$P_{j,c}(k) = P(A_{j,c}(k)|\mathbf{z}(k)) \tag{B.6}$$

Next, the target state estimates are updated as follows.

$$\hat{\mathbf{x}}^{s_i,c}(k|k) = \hat{\mathbf{x}}^{s_i,c}(k|k-1) + \mathbf{W}^{s_i,c}(k)\nu^{s_i,c}(k) \tag{B.7}$$

where

$$\begin{aligned} \nu^{s_i,c}(k) &= \sum_{i=1}^{|\bar{\mathbf{z}}^{s_i,c}(k)|} P_{i,c}(k)\mathbf{v}_j^{s_i,c}(k) \\ \mathbf{W}^{s_i,c}(k) &= \hat{\mathbf{\Sigma}}^{s_i,c}(k|k-1)\mathbf{H}(k)'\mathbf{S}^{s_i,c}(k)^{-1} \end{aligned} \tag{B.8}$$

Finally, the target covariance estimates are updated as follows.

$$\begin{aligned} \hat{\mathbf{\Sigma}}^{s_i,c}(k|k) = P_{0,c}(k)\hat{\mathbf{\Sigma}}^{s_i,c}(k|k-1)+ \\ \left[1 - P_{0,c}(k)\right]\bar{\mathbf{\Sigma}}^{s_i,c}(k|k) + \tilde{\mathbf{\Sigma}}^{s_i,c}(k|k) \end{aligned} \tag{B.9}$$

where

$$P_0(k) = 1 - \sum_{i}^{|\bar{\mathbf{z}}^{s_i,c}(k)|} P_{i,c}(k) \tag{B.10}$$

$$\bar{\mathbf{\Sigma}}^{s_i,c}(k|k) = \hat{\mathbf{\Sigma}}^{s_i,c}(k|k-1) - \mathbf{W}^{s_i,c}(k)\mathbf{S}^{s_i,c}(k)\mathbf{W}^{s_i,c}(k)' \tag{B.11}$$

$$\begin{aligned} \tilde{\mathbf{\Sigma}}^{s_i,c}(k|k) = \mathbf{W}^{s_i,c}(k)\{\sum_{i=1}^{|\bar{\mathbf{z}}^{s_i,c}(k)|} P_{i,c}(k)\mathbf{v}_j^{s_i,c}(k)\mathbf{v}_j^{s_i,c}(k)' - \\ \nu^{s_i,c}(k)\nu^{s_i,c}(k)'\}\mathbf{W}^{s_i,c}(k)' \end{aligned} \tag{B.12}$$