Chapter 10
Fully adaptive radar resource allocation for tracking and classification

Kristine Bell[1], Christopher Kreucher[2], Aaron Brandewie[3], and Joel Johnson[4]

10.1 Introduction

Modern digital radars offer unprecedented flexibility in their waveforms, radar parameter settings, and transmission schemes in order to support multiple radar system objectives including target detection, tracking, classification, and other functions. This flexibility provides the potential for improved system performance, but requires a closed-loop sense and respond approach to realize that potential. The concept of fully adaptive radar (FAR), also called cognitive radar [1–5], is to mimic the perception-action cycle (PAC) of cognition [6] to adapt the radar sensor in this closed-loop manner. In this work, we apply the FAR concept to the radar resource allocation (RRA) problem to decide how to allocate finite radar resources such as time, bandwidth, and antenna beamwidth to multiple competing radar system tasks and decide the transmission parameters for each task so that radar resources are used efficiently and system performance is optimized.

A number of perception-action approaches to RRA have been proposed, including [7–19]. Recent work in this area has been referred to as cognitive radar resource management [16–19], while older related work has been referred to as simply sensor management and/or resource allocation [7–15]. These algorithms rely on two fundamental steps. First, they capture (perceive) the state of the surveillance area probabilistically. Next, they use this probabilistic description to select future sensing actions by determining which actions are expected to maximize utility.

A key challenge of any RRA algorithm is to balance the multiple competing objectives of target detection, tracking, classification, and other radar tasks. This is addressed through the objective function used in the optimization step to select the next radar actions. Objective functions are also referred to as payoff, criteria, value, or cost functions. Articulating the system goals in a mathematical form suitable

[1]Metron, Inc., Reston, VA, USA
[2]Centauri, Ann Arbor, MI, USA
[3]The Ohio State University, Columbus, OH, USA
[4]The Ohio State University, Columbus, OH, USA

for optimization is thus critical to the operation of a fully adaptive radar resource allocation (FARRA) system. As the number of parameters available for adaptation and the number of radar system tasks grow, this becomes increasingly difficult. There are two basic approaches to this optimization: task-driven [19] and information-driven [10].

In the task-driven approach, performance quality of service (QoS) requirements are specified for each task, such as the expected time to detect a target or the tracking root-mean-squared error (RMSE), and a composite objective function is constructed by weighting the utility of various tasks. This has the benefit of being able to separately control task performance and lay out the relative importance of the tasks. However, it requires significant domain knowledge and judgment on the part of the user to specify task requirements and sensor costs and to construct cost/utility functions and weightings for combining disparate task performance metrics [19–21].

In the information-driven approach, a global information measure is optimized. Common measurements of information include entropy, mutual information (MI), Kullback-Leibler divergence (KLD), and Rényi (alpha) divergence [8, 22–25]. Information metrics implicitly balance different types of information that a radar may acquire. This has the desirable property of a common measuring stick (information flow) for all tasks [13], but does not explicitly optimize a task criterion such as RMSE. As such, the information theoretic measures can be difficult for the end-user to understand and attribute to specific operational goals [26]. Furthermore, without additional ad-hoc weighting, they do not allow for separate control of tasks and may produce solutions that over-emphasize some tasks at the expense of others or select sensor actions that provide only marginal gain when judged by user preference.

In this work, we consider a radar system performing concurrent tracking and classification of multiple targets. The FAR framework developed in [18, 27], which is based on stochastic optimization [28], provides the structure for our PAC. We develop and compare task and information-driven FARRA algorithms for allocating system resources and setting radar transmission parameters, and illustrate the performance on a simulated airborne radar scenario and on the Cognitive Radar Engineering Workspace (CREW) laboratory testbed at The Ohio State University. This work combines and extends our previous work in sensor management [8–14] and FAR [18, 21, 27, 29–31]. A preliminary version was published in [32]. The results show that the task and information-driven algorithms have similar performance but select different actions to achieve their solutions. We show that the task and information-driven algorithms are actually based on common information-theoretic quantities, so the distinction between them is in the granularity of the metrics used and the degree to which the metrics are weighted.

This chapter is organized as follows. In Section 10.2, we provide an overview of the FAR framework and in Section 10.3, we develop the multitarget multitask FARRA system model by specifing the components of the FAR framework for this problem. In Section 10.4 we describe the perceptual and executive processors that make up the FARRA PAC, including the task and information-based objective functions we employ. In Section 10.5 we provide airborne radar simulation results comparing the optimization approaches and in Section 10.6, we show CREW testbed results. Finally, Section 10.7 presents the conclusions from this effort.

10.2 Fully Adaptive Radar Framework

The FAR framework for a single PAC was developed in [18, 27] and is summarized here. A system block diagram is shown in Figure 10.1. The PAC consists of the perceptual processor and the executive processor. The PAC interacts with the external environment through the hardware sensor and with the radar system through the perceptual and executive processors. The perceptual processor receives data from the hardware sensor and processes it into a perception of the environment. The perception is passed to the radar system in order to accomplish system objectives and to the executive processor to decide the next action. The executive processor receives the perception from the perceptual processor along with requirements from the radar system, and solves an optimization problem to determine the next sensor action. The executive processor informs the hardware sensor of the settings for the next observation, the sensor collects the next set of data, and the cycle repeats.

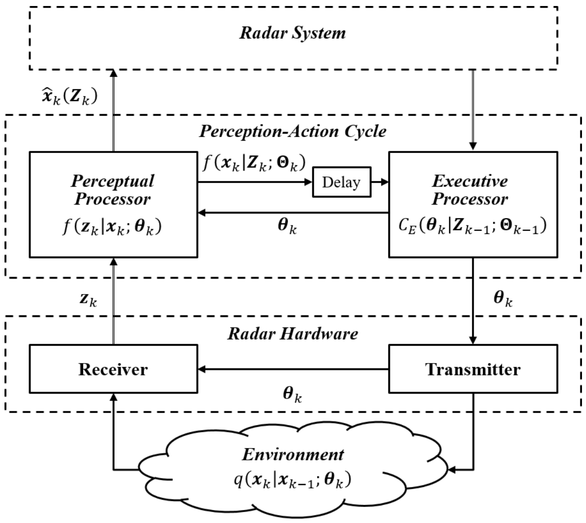


Figure 10.1: Single PAC FAR Framework

To develop the mathematical model of the PAC, we assume that the objective of the FAR system is to estimate the state of a target (or targets) at time t_k, denoted as $\boldsymbol{x}_k$. The time-varying nature of the target state is characterized by the state transition (motion) model, which is assumed to be a first order a Markov model with initial target state probability density function (PDF) $q(\boldsymbol{x}_0)$ and transition PDF $q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1};\boldsymbol{\theta}_k)$, which represents the probability that a target in state $\boldsymbol{x}_{k-1}$ will evolve to state $\boldsymbol{x}_k$. The transition density may depend on the sensor parameters $\boldsymbol{\theta}_k$; this will occur, for example, when the choice of sensor parameters affects the time difference $t_k - t_{k-1}$. The hardware sensor observes the environment and produces a measurement vector $\boldsymbol{z}_k$ that depends on the target state $\boldsymbol{x}_k$ and the sensor parameters $\boldsymbol{\theta}_k$. The measurement model is described by the conditional PDF, or likelihood function, $f(\boldsymbol{z}_k|\boldsymbol{x}_k;\boldsymbol{\theta}_k)$.

The perceptual processor processes the data and produces a perception of the target state in the form of a posterior PDF $f(\boldsymbol{x}_k|\boldsymbol{Z}_k;\boldsymbol{\Theta}_k)$ and a target state estimate $\hat{\boldsymbol{x}}_k(\boldsymbol{Z}_k)$, where $\boldsymbol{Z}_k \doteq \{\boldsymbol{z}_1,\boldsymbol{z}_2,\cdots,\boldsymbol{z}_k\}$ denotes the measurements up to time t_k and $\boldsymbol{\Theta}_k \doteq \{\boldsymbol{\theta}_1,\boldsymbol{\theta}_2,\cdots,\boldsymbol{\theta}_k\}$ denotes the sensor parameters up to time t_k. For the Markov motion model, the posterior PDF of $\boldsymbol{x}_k$ given $\boldsymbol{Z}_k$ can be obtained from the Bayes-Markov recursion:

$$\begin{aligned} f^{+}(\boldsymbol{x}_0) &= q(\boldsymbol{x}_0) && (10.1)\\ f^{-}(\boldsymbol{x}_k) \doteq f(\boldsymbol{x}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_k) &= \int q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1};\boldsymbol{\theta}_k)f^{+}(\boldsymbol{x}_{k-1})d\boldsymbol{x}_{k-1} && (10.2)\\ f^{-}(\boldsymbol{z}_k) \doteq f(\boldsymbol{z}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_k) &= \int f(\boldsymbol{z}_k|\boldsymbol{x}_k;\boldsymbol{\theta}_k)f^{-}(\boldsymbol{x}_k)d\boldsymbol{x}_k && (10.3)\\ f^{+}(\boldsymbol{x}_k) \doteq f(\boldsymbol{x}_k|\boldsymbol{Z}_k;\boldsymbol{\Theta}_k) &= \frac{f(\boldsymbol{z}_k|\boldsymbol{x}_k;\boldsymbol{\theta}_k)f^{-}(\boldsymbol{x}_k)}{f^{-}(\boldsymbol{z}_k)}, && (10.4) \end{aligned}$$

where $f^{-}(\boldsymbol{x}_k)$ is the motion-updated predicted density and $f^{+}(\boldsymbol{x}_k)$ is the information-updated posterior density. The state estimation performance is characterized by the posterior Bayes risk, which is the expected value of the perceptual processor error function $\varepsilon(\hat{\boldsymbol{x}}(\boldsymbol{Z}_k),\boldsymbol{x}_k)$ with respect to the posterior PDF,

$$R^{+}(\boldsymbol{Z}_k;\boldsymbol{\Theta}_k) = E^{+}\{\varepsilon(\hat{\boldsymbol{x}}(\boldsymbol{Z}_k),\boldsymbol{x}_k)\}, \qquad (10.5)$$

where $E_k^{+}\{\cdot\}$ denotes expectation with respect to $f^{+}(\boldsymbol{x}_k)$. The state estimate is found by minimizing the posterior Bayes risk:

$$\hat{\boldsymbol{x}}_k(\boldsymbol{Z}_k) = \underset{\hat{\boldsymbol{x}}(\boldsymbol{Z}_k)}{\arg\min}\; R^{+}(\boldsymbol{Z}_k;\boldsymbol{\Theta}_k). \qquad (10.6)$$

The goal of the executive processor is to find the next set of sensor parameters to optimize the performance of the state estimator that will include the next observation $\boldsymbol{z}_k$ as well as the previously received observations $\boldsymbol{Z}_{k-1}$. We define the joint conditional PDF of $\boldsymbol{x}_k$ and $\boldsymbol{z}_k$ conditioned on $\boldsymbol{Z}_{k-1}$ as

$$f^{\uparrow}(\boldsymbol{x}_k,\boldsymbol{z}_k) \doteq f(\boldsymbol{x}_k,\boldsymbol{z}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_k) = f^{+}(\boldsymbol{x}_k)f^{-}(\boldsymbol{z}_k). \qquad (10.7)$$

We define the predicted conditional (PC)-Bayes risk by taking the expectation of the error function with respect to the joint conditional PDF,

$$R^{\uparrow}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}) = E_k^{\uparrow}\{\varepsilon(\hat{\boldsymbol{x}}(\boldsymbol{Z}_k),\boldsymbol{x}_k)\}, \qquad (10.8)$$

where $E_k^{\uparrow}\{\cdot\}$ denotes expectation with respect to $f^{\uparrow}(\boldsymbol{x}_k,\boldsymbol{z}_k)$. We can also write the PC-Bayes risk as the expectation of the posterior Bayes risk with respect to $f^{-}(\boldsymbol{z}_k)$, i.e.,

$$R^{\uparrow}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}) = E_{\boldsymbol{z}_k}^{-}\{R^{+}(\boldsymbol{Z}_k;\boldsymbol{\Theta}_k)\}, \qquad (10.9)$$

where $E_{\boldsymbol{z}_k}^{-}\{\cdot\}$ denotes expectation with respect to $f^{-}(\boldsymbol{z}_k)$. In many applications, the PC-Bayes risk may be difficult to compute and in general will not have a closed form analytical expression. To overcome this difficulty, information-theoretic surrogate functions that are analytically tractable and provide a good indication of the quality of the target state estimate are often substituted. The next set of sensor parameters are chosen to minimize an executive cost (or objective) function $C_E(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1})$. In the task-driven approach, the executive cost function is a scalar function that incorporates the processor performance, derived from the PC-Bayes risk or a surrogate, with system requirements and the cost of obtaining measurements. In the information-driven approach, the executive cost function is an information theoretic measure. The executive processor optimization problem is then

$$\boldsymbol{\theta}_k = \arg\min_{\boldsymbol{\theta}} C_E(\boldsymbol{\theta}|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}). \tag{10.10}$$

In the next two sections, we specialize the general FAR framework for the multitarget multitask RRA problem.

10.3 Multitarget Multitask FARRA System Model

The multitarget multitask FARRA system model is shown in Figure 10.2. There is a single PAC with a perceptual processor that consists of M tasks and an executive processor that allocates system resources to the M tasks and specifies the next sequence of transmissions of the radar.

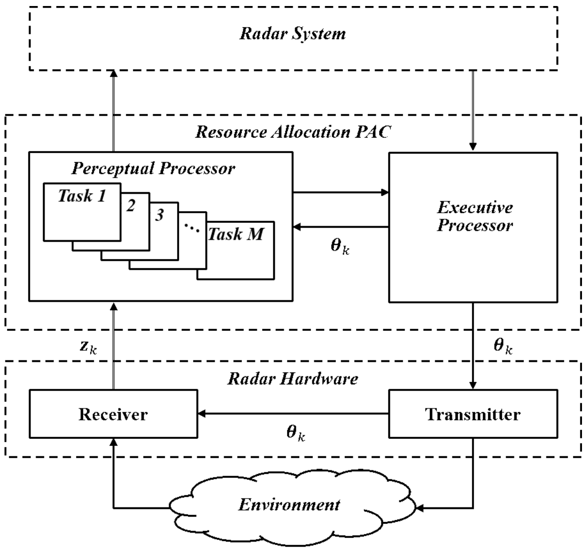


Figure 10.2: System Model for Multiple Task FARRA

10.3.1 Radar Resource Allocation Model

We define a resource allocation frame as an interval of fixed length T_F and let k denote the frame (time) index. We assume there are M variable length dwells in the kth frame, corresponding to M different tasks, where M is fixed and known. During each task dwell, we assume that the radar can transmit nothing (taking up no time) or one of L waveforms from a waveform library. Let $a_l; l = 0, \ldots, L$ denote each of the possible actions (waveforms), where a_0 is the action of no transmission, and let $A = \{a_0, \ldots, a_L\}$ denote the set of actions.

10.3.2 Controllable Parameters

The radar resource parameter vector, or action vector, for the kth frame is defined as the $M \times 1$ vector

$$\boldsymbol{\theta}_k = \begin{bmatrix} \theta_{1,k} & \theta_{2,k} & \cdots & \theta_{M,k} \end{bmatrix}^T, \tag{10.11}$$

where $\theta_{m,k} \in A$ is the action in the mth dwell of the kth frame. The objective of the FARRA executive processor is to determine the best action vector for the next one or more frames.

10.3.3 State Vector

Following TBD, we assume there are N targets, where N is fixed and known, and the multitarget state vector has the form:

$$\boldsymbol{x}_k = \begin{bmatrix} \boldsymbol{x}_{1,k}^T & \boldsymbol{x}_{2,k}^T & \cdots & \boldsymbol{x}_{N,k}^T \end{bmatrix}^T, \tag{10.12}$$

where $\boldsymbol{x}_{n,k}$ is the state vector for the nth target and contains components relevant to the tasks at hand. Here we consider multitarget tracking and classification, therefore each target's state is composed of a tracking state vector $\boldsymbol{y}_{n,k}$ and a classification state variable $c_{n,k}$:

$$\boldsymbol{x}_{n,k} = \begin{bmatrix} \boldsymbol{y}_{n,k}^T & c_{n,k} \end{bmatrix}^T. \tag{10.13}$$

The tracking state vector $\boldsymbol{y}_{n,k}$ consists of kinematic variables (position, velocity, and possibly acceleration) and the received signal-to-noise ratio (SNR). The tracking state variables are continuous random variables, while the target class is a discrete random variable that takes on one of a discrete set of values. As such, the various PDFs define in Section 10.2 become a combination of a PDF for the continuous components and a probability mass function (PMF) for the discrete components.

10.3.4 Transition Model

The transition model consists of a prior PDF/PMF $q(\boldsymbol{x}_0)$ and a transition PDF/PMF $q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}; \boldsymbol{\theta}_k)$. We assume the target transition models are independent across targets, and that for each target, the tracking and classification transition models are independent, therefore the joint tracking and classification transition model has the form:

$$q(\boldsymbol{x}_0) = \prod_{n=1}^{N} q(\boldsymbol{x}_{n,0}) = \prod_{n=1}^{N} q(\boldsymbol{y}_{n,0}) q(c_{n,0}) \tag{10.14}$$

and

$$q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) = \prod_{n=1}^{N} q(\boldsymbol{x}_{n,k}|\boldsymbol{x}_{n,k-1}; \boldsymbol{\theta}_k) = \prod_{n=1}^{N} q(\boldsymbol{y}_{n,k}|\boldsymbol{y}_{n,k-1}; \boldsymbol{\theta}_k) q(c_{n,k}|c_{n,k-1}). \tag{10.15}$$

In this model we assume that the tracking transition model depends on the sensor parameters but the classification transition model does not, as explained below.

For each target, we assume an initial tracking state distribution of the form $\boldsymbol{y}_{n,0} \sim N\left(\boldsymbol{\mu}_{n,0}, \boldsymbol{\Sigma}_{n,0}\right)$, where the notation $\boldsymbol{y} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ means that the random vector $\boldsymbol{y}$ has a multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. We use the notation $N(\boldsymbol{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ to denote the multivariate Gaussian PDF for the random variable $\boldsymbol{y}$, i.e.

$$N(\boldsymbol{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \doteq \frac{1}{\sqrt{|2\pi\boldsymbol{\Sigma}|}} \exp\left\{ -\frac{1}{2} [\boldsymbol{y} - \boldsymbol{\mu}]^T \boldsymbol{\Sigma}^{-1} [\boldsymbol{y} - \boldsymbol{\mu}] \right\}. \tag{10.16}$$

Therefore

$$q(\boldsymbol{y}_{n,0}) = N\left(\boldsymbol{y}_{n,0}; \boldsymbol{\mu}_{n,0}, \boldsymbol{\Sigma}_{n,0}\right). \tag{10.17}$$

Let $\Delta t(\boldsymbol{\theta}_k) = t_k - t_{k-1}$. We assume a linear, additive white Gaussian noise (AWGN) motion model of the form:

$$\boldsymbol{y}_{n,k} = \boldsymbol{F}_n\left(\Delta t(\boldsymbol{\theta}_k)\right) \boldsymbol{y}_{n,k-1} + \boldsymbol{e}_{n,k}, \tag{10.18}$$

where the process noise distribution is $\boldsymbol{e}_{n,k} \sim N\left(\boldsymbol{0}, \boldsymbol{Q}_n\left(\Delta t(\boldsymbol{\theta}_k)\right)\right)$ so that the transition PDF is:

$$q(\boldsymbol{y}_{n,k}|\boldsymbol{y}_{n,k-1}; \boldsymbol{\theta}_k) = N\left(\boldsymbol{y}_{n,k}; \boldsymbol{F}_n\left(\Delta t(\boldsymbol{\theta}_k)\right) \boldsymbol{y}_{n,k-1}, \boldsymbol{Q}_n\left(\Delta t(\boldsymbol{\theta}_k)\right)\right). \tag{10.19}$$

We assume the target class $c_{n,k}$ takes on one of a discrete set of N_c values in the set C,

$$c_{n,k} \in C = \{1, 2, \ldots, N_c\}. \tag{10.20}$$

For each target, the prior distribution is characterized by the PMF $q(c_{n,0})$, which is represented by the $N_c \times 1$ vector $\boldsymbol{q}_n$, which consists of the N_c probabilities

$$[\boldsymbol{q}_n]_i = P(c_{n,0} = i); \qquad i = 1, \ldots, N_c. \tag{10.21}$$

The transition model $q(c_{n,k}|c_{n,k-1})$ is represented by the $N_c \times N_c$ transition matrix $\boldsymbol{\Upsilon}_n$, where

$$[\boldsymbol{\Upsilon}_n]_{ij} = P(c_{n,k} = i | c_{n,k-1} = j); \qquad i, j = 1, \ldots, N_c. \tag{10.22}$$

Depending on the model, the target may or may not be able to switch classes. For example, if the class represents a target behavior class, then switching can occur, however if the class represents a type of vehicle or aircraft, then switching cannot occur and $\boldsymbol{\Upsilon}_n = \boldsymbol{I}$. We assume that the classification transition model does not depend on the time between updates or the sensor parameters $\boldsymbol{\theta}_k$.

10.3.5 Measurement Model

We assume that each target is allocated one dwell per frame, in which up to one tracking and up to one classification measurement is obtained. Let $\boldsymbol{z}_{n,k}$ and $\boldsymbol{\xi}_{n,k}$ denote the tracking and classification measurement vectors, respectively, for the nth target during the kth frame. Either or both of these may be empty if there is no measurement of that type. Thus we have $2N$ measurements from $M = N$ dwells and the measurement vector has the form:

$$\boldsymbol{z}_k = \begin{bmatrix} \boldsymbol{z}_{1,k}^T & \boldsymbol{\xi}_{1,k}^T & \boldsymbol{z}_{2,k}^T & \boldsymbol{\xi}_{2,k}^T & \cdots & \boldsymbol{z}_{N,k}^T & \boldsymbol{\xi}_{N,k}^T \end{bmatrix}^T. \tag{10.23}$$

We assume the measurements are independent, thus the likelihood function has the form:

$$f(\boldsymbol{z}_k|\boldsymbol{x}_k;\boldsymbol{\theta}_k) = \prod_{n=1}^{N} f(\boldsymbol{z}_{n,k}|\boldsymbol{y}_{n,k};\boldsymbol{\theta}_k) f(\boldsymbol{\xi}_{n,k}|c_{n,k};\boldsymbol{\theta}_k). \tag{10.24}$$

For tracking, we assume that measurements are received with detection probability $P_D\left(\boldsymbol{y}_{n,k};\boldsymbol{\theta}_k\right)$, where the detection probability is determined by the received SNR, which is a function of the target state and sensor parameters. When measurements are received, we assume they follow a nonlinear, AWGN measurement model of the form:

$$\boldsymbol{z}_{n,k} = \boldsymbol{h}_n\left(\boldsymbol{y}_{n,k}\right) + \boldsymbol{n}_{n,k}, \tag{10.25}$$

where $\boldsymbol{h}_n\left(\boldsymbol{y}_{n,k}\right)$ is a nonlinear transformation from the target state space to the radar measurement space and $\boldsymbol{n}_{n,k}$ is the measurement error, which is modeled as $\boldsymbol{n}_{n,k} \sim N\left(\boldsymbol{0}, \boldsymbol{R}_{n,k}\left(\boldsymbol{\theta}_k\right)\right)$, where $\boldsymbol{R}_{n,k}\left(\boldsymbol{\theta}_k\right)$ is the measurement covariance matrix. The single target tracking likelihood function is then:

$$f(\boldsymbol{z}_{n,k}|\boldsymbol{y}_{n,k};\boldsymbol{\theta}_k) = N\left(\boldsymbol{z}_{n,k};\boldsymbol{h}_n\left(\boldsymbol{y}_{n,k}\right), \boldsymbol{R}_{n,k}\left(\boldsymbol{\theta}_k\right)\right). \tag{10.26}$$

For classification, we consider two measurement models: a discrete class measurement model and a continuous Gaussian feature vector model. In the discrete class model, we assume that the sensor makes a discrete valued measurement of target class, i.e. $\xi_{n,k} \in C = \{1, 2, \ldots, N_c\}$ and the likelihood function $f(\xi_{n,k}|c_{n,k};\boldsymbol{\theta}_k)$ is represented by the $N_c \times N_c$ likelihood matrix $\boldsymbol{L}_n(\boldsymbol{\theta}_k)$, where

$$[\boldsymbol{L}_n]_{ij}\left(\boldsymbol{\theta}_k\right) = P(\xi_{n,k} = i|c_{n,k} = j); \qquad i, j = 1, \ldots, N_c. \tag{10.27}$$

In the Gaussian feature vector model, we assume that the sensor makes a continuous valued measurement of a feature vector $\boldsymbol{\xi}_{n,k}$ and the likelihood function is a Gaussian density with mean and covariance matrix determined by the target class. For the ith class, the likelihood function is:

$$f(\boldsymbol{\xi}_{n,k}|c_{n,k} = i;\boldsymbol{\theta}_k) = N\left(\boldsymbol{\xi}_{n,k};\boldsymbol{\mu}_{n,i}\left(\boldsymbol{\theta}_k\right), \boldsymbol{\Sigma}_{n,i}\left(\boldsymbol{\theta}_k\right)\right); \quad i = 1, \ldots, N_c. \tag{10.28}$$

10.4 FARRA PAC

10.4.1 Perceptal Processor

Since the motion and measurement models developed in Section 10.3 are independent across targets and tasks, the Bayes-Markov recursions decouple and can be computed separately for the tracking and classification tasks for each target. While the Bayes-Markov recursion expressions in (10.1)-(10.4) appear straightforward, they are usually analytically and/or computationally infeasible to evaluate exactly. One exception is in the tracking problem when the motion and measurement models are linear with AWGN, and the transition density, likelihood function, predicted density, and posterior density are all Gaussian. In this case the exact solution is given by the linear Kalman filter (KF) and the motion and measurement updates consist of explicit linear calculations of the mean vectors and covariance matrices that characterize the Gaussian densities. For the general tracking problem, approximate and suboptimal implementations include the extended Kalman filter (EKF), unscented Kalman filter (UKF), particle filters, and many others. Our tracking model includes a nonlinear AWGN measurement model and we will use the EKF as an approximate solution to the Bayes-Markov recursion. The EKF reduces to the exact KF if the measurement model is in fact linear. Another exception is in the classification problem when the target state is one of a discrete set of classes, the motion model is specified by a transition matrix, and the likelihood function has a closed form analytical expression. Our classification models meet these requirements.

For the tracking tasks, the predicted and posterior PDFs are computed using the EKF. The PDFs are presumed to be Gaussian of the form

$$f^{-}(\boldsymbol{y}_{n,k}) = N\left(\boldsymbol{y}_{n,k}; \boldsymbol{\mu}^{-}_{n,k}, \boldsymbol{P}^{-}_{n,k}\right) \tag{10.29}$$

$$f^{+}(\boldsymbol{y}_{n,k}) = N\left(\boldsymbol{y}_{n,k}; \boldsymbol{\mu}^{+}_{n,k}, \boldsymbol{P}^{+}_{n,k}\right). \tag{10.30}$$

The EKF is initialized with:

$$\boldsymbol{\mu}^{+}_{n,0} = \boldsymbol{\mu}_{n,0} \tag{10.31}$$

$$\boldsymbol{P}^{+}_{n,0} = \boldsymbol{\Sigma}_{n,0} \tag{10.32}$$

and the recursions have the from:

$$\boldsymbol{\mu}^{-}_{n,k} = \boldsymbol{F}_n\left(\Delta t(\boldsymbol{\theta}_k)\right)\boldsymbol{\mu}^{+}_{n,k-1} \tag{10.33}$$

$$\boldsymbol{P}^{-}_{n,k} = \boldsymbol{F}_n\left(\Delta t(\boldsymbol{\theta}_k)\right)\boldsymbol{P}^{+}_{n,k-1}\boldsymbol{F}_n\left(\Delta t(\boldsymbol{\theta}_k)\right)^T + \boldsymbol{Q}_n\left(\Delta t(\boldsymbol{\theta}_k)\right) \tag{10.34}$$

$$\boldsymbol{H}_{n,k} = \boldsymbol{H}_n(\boldsymbol{\mu}^{-}_{n,k}) \tag{10.35}$$

$$\boldsymbol{K}_{n,k} = \boldsymbol{P}^{-}_{n,k}\boldsymbol{H}^{T}_{n,k}\left[\boldsymbol{H}_{n,k}\boldsymbol{P}^{-}_{n,k}\boldsymbol{H}^{T}_{n,k} + \boldsymbol{R}_{n,k}(\boldsymbol{\theta}_k)\right]^{-1} \tag{10.36}$$

$$\boldsymbol{\mu}^{+}_{n,k} = \boldsymbol{\mu}^{-}_{n,k} + \boldsymbol{K}_{n,k}\left[\boldsymbol{z}_{n,k} - \boldsymbol{h}_n(\boldsymbol{\mu}^{-}_{n,k})\right] \tag{10.37}$$

$$\boldsymbol{P}^{+}_{n,k} = \boldsymbol{P}^{-}_{n,k} - \boldsymbol{K}_{n,k}\boldsymbol{H}_{n,k}\boldsymbol{P}^{-}_{n,k}, \tag{10.38}$$

where $\boldsymbol{H}_n(\boldsymbol{y})$ is the Jacobian matrix, defined as:

$$\boldsymbol{H}_n(\boldsymbol{y}) = \left[\nabla_{\boldsymbol{y}} \boldsymbol{h}_n(\boldsymbol{y})^T\right]^T . \tag{10.39}$$

For the classification tasks, the predicted and posterior PMFs are computed using the exact Bayes-Markov recursions. Let $f_i^-(c_{n,k}) \doteq P(c_{n,k} = i|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1})$ denote the predicted PMF and $f_i^+(c_{n,k}) \doteq P(c_{n,k} = i|\boldsymbol{Z}_k;\boldsymbol{\Theta}_k)$ denote the posterior PMF. The recursion is initialized with:

$$f_i^+(c_{n,0}) = [\boldsymbol{q}_n]_i; \qquad i = 1,\ldots,N_c, \tag{10.40}$$

and the predicted PMF is computed from:

$$f_i^-(c_{n,k}) = \sum_{j=1}^{N_c} [\boldsymbol{\Upsilon}_n]_{ij} f_j^+(c_{n,k-1}); \qquad i = 1,\ldots,N_c. \tag{10.41}$$

For the discrete class measurement model, the information update has the form:

$$f^-(\xi_{n,k}) = \sum_{j=1}^{N_c} [\boldsymbol{L}_n]_{\xi_{n,k},j}(\boldsymbol{\theta}_k) f_j^-(c_{n,k}) \tag{10.42}$$

$$f_i^+(c_{n,k}) = \frac{[\boldsymbol{L}_n]_{\xi_{n,k},i}(\boldsymbol{\theta}_k) f_i^-(c_{n,k})}{f^-(\xi_{n,k})}; \quad i = 1,\ldots,N_c, \tag{10.43}$$

while for the Gaussian feature vector measurement model, the information update has the form:

$$f^-(\boldsymbol{\xi}_{n,k}) = \sum_{j=1}^{N_c} f(\boldsymbol{\xi}_{n,k}|c_{n,k} = j;\boldsymbol{\theta}_k) f_j^-(c_{n,k}) \tag{10.44}$$

$$f_i^+(c_{n,k}) = \frac{f(\boldsymbol{\xi}_{n,k}|c_{n,k} = i;\boldsymbol{\theta}_k) f_i^-(c_{n,k})}{f^-(\boldsymbol{\xi}_{n,k})}; \quad i = 1,\ldots,N_c. \tag{10.45}$$

The posterior Bayes risk for the multitarget tracking and classification state vector is the sum of the the traces of the posterior mean square error (MSE) matrices and the posterior probability of incorrect classification across all targets. The solution to (10.6) is the mean of the posterior PDF for the tracking variables and the maximum of the posterior PMF for the classification variables:

$$\hat{\boldsymbol{y}}_{n,k}(\boldsymbol{Z}_k) = E_k^+\left\{\boldsymbol{y}_{n,k}\right\} = \boldsymbol{\mu}_{n,k}^+; \qquad n = 1,\ldots,N \tag{10.46}$$

$$\hat{c}_{n,k}(\boldsymbol{Z}_k) = \arg\max_{i\in C} f_i^+\left(c_{n,k}\right); \qquad n = 1,\ldots,N. \tag{10.47}$$

10.4.2 Executive Processor

In this chapter, we develop both task-driven and information-driven methods for specifying the objective function used by the executive processor. It should be noted here that in both approaches, the optimization is in a global sense and may not be the optimal solution for a particular radar task.

10.4.2.1 Task-Driven (QoS) Approach

Following the development in [19], we assume there are M tasks. The perceptual processor for the mth task computes a perception of its environment, which may include quantities such as target location, target class, target SNR, etc. The executive processor for the mth task analytically evaluates the performance of the perceptual processor in terms of a task QoS metric, which is denoted by $G_{m,k}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1})$. The QoS metric for the current frame will in general depend on the perception from the previous frame, the previous sensing actions, and the current sensing action. Each task QoS metric has a task QoS requirement, denoted $\bar{G}_m$. The task QoS metrics and requirements are physically meaningful quantities with appropriate physical units. The task QoS metric and requirement are converted to a task utility $U_{m,k}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1})$, which is a unitless quantity on the interval $[0,1]$. It represents the level of satisfaction with the QoS and is determined from the task utility function,

$$U_{m,k}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}) = u_m(G_{m,k}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}),\bar{G}_m). \tag{10.48}$$

The executive processor combines and balances the task utilities along with resource constraints to determine the resource allocation for the next frame. The mission utility, or mission effectiveness, is a measure of the radar system's ability to meet all of its requirements. It is a weighted sum of the task utilities, where the task weighting, w_m, represents the relative importance of the mth task to the overall mission, and the weights sum to one. The mission utility is given by

$$U(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1})) = \sum_{m=1}^{M} w_m U_{m,k}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}). \tag{10.49}$$

Constraints on system resources are described by the function $g_c(\boldsymbol{\theta}_k)$, constructed so the constraint may be expressed as the inequality $g_c(\boldsymbol{\theta}_k) \leq 0$. The next action vector is then determined by maximizing the mission utility subject to the constraint

$$\boldsymbol{\theta}_k = \arg\max_{\boldsymbol{\theta}} U(\boldsymbol{\theta}|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}), \qquad s.t. g_c(\boldsymbol{\theta}) \leq 0. \tag{10.50}$$

For a tracking task, we use the position and velocity RMSE and the requirement is an upper limit on the RMSE. In most cases, it is not possible to evaluate the RMSE analytically. However, the Bayesian Cramér-Rao lower bound (BCRLB), the inverse of the Bayesian information matrix (BIM), provides a (matrix) lower bound on the MSE matrix of any estimator [33] and is usually analytically tractable. For tracking applications, this yields the posterior Cramér-Rao lower bound (PCRLB) [34]. The PCRLB provides a lower bound on the global MSE that has been averaged over $\boldsymbol{x}_k$ and $\boldsymbol{Z}_k$, thus it characterizes tracker performance for all possible data that might have been received. Here we use a predicted conditional BIM (PC-BIM) and a predicted conditional Cramér-Rao lower bound (PC-CRLB) to bound the PC-MSE matrix, which is averaged over the joint density of $\boldsymbol{x}_k$ and $\boldsymbol{z}_k$ conditioned on $\boldsymbol{Z}_{k-1}$. The PC-CRLB differs from the PCRLB in that it characterizes performance conditioned on the actual data that has been received. For our model the PC-BIM has the same form as the inverse of the EKF posterior covariance matrix in (10.38), which simplifies to

$$\begin{aligned} \boldsymbol{B}^{\uparrow}_{n,k}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}) &= \left[\boldsymbol{P}^{-}_{n,k}-\boldsymbol{K}_{n,k}\boldsymbol{H}_{n,k}\boldsymbol{P}^{-}_{n,k}\right]^{-1} \\ &= \left[\boldsymbol{P}^{-}_{n,k}\right]^{-1}+\boldsymbol{H}^{T}_{n,k}\boldsymbol{R}_{n,k}(\boldsymbol{\theta}_k)^{-1}\boldsymbol{H}_{n,k}. \end{aligned} \tag{10.51}$$

In our model, a detection is obtained with probability $P_D\left(\boldsymbol{y}_{n,k};\boldsymbol{\theta}_k\right)$. When a detection is obtained, the PC-BIM has the form given in (10.51). When a detection is missed, the second term is equal to zero and the PC-BIM is equal to the inverse of the predicted covariance matrix. Using the approach of the information reduction factor bound in [35], and substituting the mean of the predicted density for the unknown target state, the tracking PC-BIM with missed detections is:

$$\tilde{\boldsymbol{B}}^{\uparrow}_{n,k}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1})=\left[\boldsymbol{P}^{-}_{n,k}\right]^{-1}+P_D\left(\boldsymbol{\mu}^{-}_{n,k};\boldsymbol{\theta}_k\right)\boldsymbol{H}^{T}_{n,k}\boldsymbol{R}_{n,k}(\boldsymbol{\theta}_k)^{-1}\boldsymbol{H}_{n,k}. \tag{10.52}$$

The tracking PC-CRLB is the inverse of the PC-BIM,

$$\tilde{\boldsymbol{C}}^{\uparrow}_{n,k}(\boldsymbol{\theta}_k)=\tilde{\boldsymbol{B}}^{\uparrow}_{n,k}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1})^{-1}, \tag{10.53}$$

where we have temporarily dropped the conditioning on $\boldsymbol{Z}_{k-1}$ and $\boldsymbol{\Theta}_{k-1}$ to simplify the notation. The QoS metrics are the position and velocity RMSEs obtained from the PC-CRLB as follows:

$$G^{R}_{n,k}(\boldsymbol{\theta}_k) = \sqrt{\left[\tilde{\boldsymbol{C}}^{\uparrow}_{n,k}(\boldsymbol{\theta}_k)\right]_x+\left[\tilde{\boldsymbol{C}}^{\uparrow}_{n,k}(\boldsymbol{\theta}_k)\right]_y+\left[\tilde{\boldsymbol{C}}^{\uparrow}_{n,k}(\boldsymbol{\theta}_k)\right]_z} \tag{10.54}$$

$$G^{V}_{n,k}(\boldsymbol{\theta}_k) = \sqrt{\left[\tilde{\boldsymbol{C}}^{\uparrow}_{n,k}(\boldsymbol{\theta}_k)\right]_{\dot{x}}+\left[\tilde{\boldsymbol{C}}^{\uparrow}_{n,k}(\boldsymbol{\theta}_k)\right]_{\dot{y}}+\left[\tilde{\boldsymbol{C}}^{\uparrow}_{n,k}(\boldsymbol{\theta}_k)\right]_{\dot{z}}}. \tag{10.55}$$

The QoS requirements are the values that we want the RMSEs to be below, denoted as $\bar{G}^{R}_{n}$ and $\bar{G}^{V}_{n}$. We then define the position and velocity task utility functions to be:

$$U^{R}_{n,k}(\boldsymbol{\theta}_k) = \begin{cases} \dfrac{\bar{G}^{R}_{n}}{G^{R}_{n,k}(\boldsymbol{\theta}_k)} & G^{R}_{n,k}(\boldsymbol{\theta}_k)>\bar{G}^{R}_{n} \\ 1 & G^{R}_{n,k}(\boldsymbol{\theta}_k)\le\bar{G}^{R}_{n} \end{cases} \tag{10.56}$$

$$U^{V}_{n,k}(\boldsymbol{\theta}_k) = \begin{cases} \dfrac{\bar{G}^{R}_{n}}{G^{V}_{n,k}(\boldsymbol{\theta}_k)} & G^{V}_{n,k}(\boldsymbol{\theta}_k)>\bar{G}^{V}_{n} \\ 1 & G^{V}_{n,k}(\boldsymbol{\theta}_k)\le\bar{G}^{V}_{n}. \end{cases} \tag{10.57}$$

With these utility functions, if the QoS metric is below the required value, the resulting utility is one and there is neither a penalty nor any additional utility for being below the requirement.

For a classification task, the desired QoS metric is the probability of incorrect classification. The posterior probability of incorrect classification is difficult to compute and in general does not have a closed form analytical expression. The PC-probability of incorrect classification is even more difficult to compute since it involves an additional expectation over the next measurement $\boldsymbol{\xi}_{n,k}$. To overcome this difficulty, we substitute an information-theoretic surrogate that is analytically tractable and provides a good indication of the quality of the target class estimate.

As in [30–32], we use the entropy, which can be calculated directly from the discrete classification PMF. The entropy of the predicted and posterior PMFs, respectively, are defined as [22]:

$$H^-(c_{n,k}) = -\sum_{i=1}^{N_c} f_i^-(c_{n,k}) \ln f_i^-(c_{n,k}) \tag{10.58}$$

$$H^+(c_{n,k}) = -\sum_{i=1}^{N_c} f_i^+(c_{n,k}) \ln f_i^+(c_{n,k}). \tag{10.59}$$

The entropy has the property $0 \leq H \leq \ln(N_c)$. It is low when the PMF is concentrated on one of the classes and high when the PMF is distributed across the classes. The minimum value is obtained when all the probability is in one class and the maximum value is obtained when all classes have the same probability. The posterior entropy is a surrogate for the posterior probability of incorrect classification and is used to characterize classification performance after the measurement is received. In order for the executive processor to determine the next sensing action, we also need a surrogate for the PC-probability of incorrect classification, which characterizes the expected performance of the current (next) measurement, given the past measurements that have been observed. If we take the expected value of the posterior entropy with respect to $f^-(\boldsymbol{\xi}_{n,k})$, we obtain the desired surrogate, which is the conditional entropy [22] of $c_{n,k}$ given $\boldsymbol{\xi}_{n,k}$ conditioned on the past measurements $\boldsymbol{Z}_{k-1}$.

For the discrete class measurement model, we must compute $f^-(\xi_{n,k})$ and $f_i^+(c_{n,k})$ for every $\xi_{n,k}$ using (10.42) and (10.43). Using slightly more explicit notation, define $f_{i|j}^+(c_{n,k}|\xi_{n,k}) \doteq P(c_{n,k} = i|\xi_{n,k} = j, \boldsymbol{Z}_{k-1}; \boldsymbol{\Theta}_k)$ and $f_j^-(\xi_{n,k}) \doteq P(\xi_{n,k} = j|\boldsymbol{Z}_{k-1}; \boldsymbol{\Theta}_k)$. The conditional entropy is then computed from:

$$H_n^{\uparrow}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1}; \boldsymbol{\Theta}_{k-1}) = \sum_{j=1}^{N_c} f_j^-(\xi_{n,k}) \left\{ -\sum_{i=1}^{N_c} f_{i|j}^+(c_{n,k}|\xi_{n,k}) \ln f_{i|j}^+(c_{n,k}|\xi_{n,k}) \right\}. \tag{10.60}$$

For the Gaussian feature vector measurement model, we must compute $f^-(\boldsymbol{\xi}_{n,k})$ and $f_i^+(c_{n,k})$ as a function of $\boldsymbol{\xi}_{n,k}$ using (10.44) and (10.45). Using the notation $f_i^+(c_{n,k}|\boldsymbol{\xi}_{n,k}) \doteq P(c_{n,k} = i|\boldsymbol{\xi}_{n,k}, \boldsymbol{Z}_{k-1}; \boldsymbol{\Theta}_k)$, the conditional entropy is then computed from:

$$H_n^{\uparrow}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1}; \boldsymbol{\Theta}_{k-1}) = \int f^-(\boldsymbol{\xi}_{n,k}) \left\{ -\sum_{i=1}^{N_c} f_i^+(c_{n,k}|\boldsymbol{\xi}_{n,k}) \ln f_i^+(c_{n,k}|\boldsymbol{\xi}_{n,k}) d\boldsymbol{\xi}_{n,k} \right\}. \tag{10.61}$$

The integral does not have a closed form expression and must be evaluated numerically or approximated.

The QoS classification accuracy metric is the conditional entropy, $G_{n,k}^C(\boldsymbol{\theta}_k) = H_n^{\uparrow}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1}; \boldsymbol{\Theta}_{k-1})$ and the QoS requirement is the value that FARRA wants the conditional entropy to be below, denoted as $\bar{G}_n^C$. We then define the classification task utility function to be:

$$U_{n,k}^C(\boldsymbol{\theta}_k) = \begin{cases} \dfrac{\bar{G}_n^C}{G_{n,k}^C(\boldsymbol{\theta}_k)} & G_{n,k}^C(\boldsymbol{\theta}_k) > \bar{G}_n^C \\ 1 & G_{n,k}^C(\boldsymbol{\theta}_k) \leq \bar{G}_n^C \end{cases} \tag{10.62}$$

The mission utility function is obtained by assigning weights to the task utility functions, which we denote as w_n^R, w_n^V, and w_n^C, and computing the weighted sum of the task utilities,

$$U(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1})) = \sum_{n=1}^{N} \left(w_n^R U_{n,k}^R(\boldsymbol{\theta}_k) + w_n^V U_{n,k}^V(\boldsymbol{\theta}_k) + w_n^C U_{n,k}^C(\boldsymbol{\theta}_k)\right). \quad (10.63)$$

10.4.2.2 Information-Driven Approach

In the information-driven approach, the relative merit of different sensing actions is measured by the corresponding expected gain in information [8, 10, 12, 17, 25]. Assume, temporarily, that at time t_k a FARRA strategy has selected action $\boldsymbol{\theta}_k$, it has been executed, and measurement $\boldsymbol{z}_k$ has been received. To judge the value of this action, we compute the information gained by that measurement; specifically the information gain between the predicted PDF on target state before the measurement was taken $f^-(\boldsymbol{x}_k)$ and the posterior PDF after the measurement has been received $f^+(\boldsymbol{x}_k)$. The most popular approach uses the KLD. The KLD between $f^+(\boldsymbol{x}_k)$ and $f^-(\boldsymbol{x}_k)$ is defined as [22]:

$$D(f^+(\boldsymbol{x}_k)||f^-(\boldsymbol{x}_k)) \doteq \int f^+(\boldsymbol{x}_k) \ln \frac{f^+(\boldsymbol{x}_k)}{f^-(\boldsymbol{x}_k)} d\boldsymbol{x}_k. \quad (10.64)$$

There are a number of generalizations of the KLD in the literature, including the Renyi divergence, the Arimoto-divergences, and the f-divergence [23–25]. The KLD has a number of nice theoretical and practical properties, including (a) the ability to compare actions which generate different types of knowledge (e.g., knowledge about target class versus knowledge about target position) using a common measuring stick – information gain; (b) the asymptotic connection between information gain and risk-based optimization; and (c) the avoidance of weighting schemes to value different types of information. Taking the expectation with respect to $f^-(\boldsymbol{z}_k)$, we obtain the expected KLD, which is also known as the MI [22]:

$$I_{\boldsymbol{xz}}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}) \doteq E_{\boldsymbol{z}_k}^{-}\left\{\int f^+(\boldsymbol{x}_k) \ln \frac{f^+(\boldsymbol{x}_k)}{f^-(\boldsymbol{x}_k)} d\boldsymbol{x}_k\right\}. \quad (10.65)$$

The next action vector is then determined by maximizing the mutual information,

$$\boldsymbol{\theta}_k = \arg\max_{\boldsymbol{\theta}} I_{\boldsymbol{xz}}(\boldsymbol{\theta}|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}) \quad (10.66)$$

For our model, the global MI decomposes into the sum of the tracking and classification MIs, which we denote as $I_{\boldsymbol{yz};n}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1})$ and $I_{c\boldsymbol{\xi};n}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1})$, respectively,

$$I_{\boldsymbol{xz}}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}) = \sum_{n=1}^{N} \left(I_{\boldsymbol{yz};n}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}) + I_{c\boldsymbol{\xi};n}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1})\right). \quad (10.67)$$

The tracking MI has the form

$$I_{\boldsymbol{yz};n}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}) = \frac{1}{2}\ln|\boldsymbol{P}_{n,k}^-| + \frac{1}{2}\ln\left|\left[\boldsymbol{P}_{n,k}^-\right]^{-1} + \boldsymbol{H}_{n,k}^T\boldsymbol{R}_{n,k}(\boldsymbol{\theta}_k)^{-1}\boldsymbol{H}_{n,k}\right|. \quad (10.68)$$

The classification MI is the difference between the entropy of the predicted PMF in (10.58) and the conditional entropy in (10.60) or (10.61),

$$I_{c\boldsymbol{\xi};n}(\boldsymbol{\theta}_k|\mathbf{Z}_{k-1};\boldsymbol{\Theta}_{k-1}) = H^{-}(c_{n,k}) - H_n^{\uparrow}(\boldsymbol{\theta}_k|\mathbf{Z}_{k-1};\boldsymbol{\Theta}_{k-1}). \tag{10.69}$$

Comparing the second term in (10.68) to the expression for the PC-BIM in (10.51), we see that the tracking MI is a function of the determinant of the PC-BIM. Thus, the task-based and information-based methods developed here have at their core the same information theoretic quantities, and the distinction is in the separation and weighting of individual tasks in the task-based method versus a global approach in the information-based method.

10.5 Simulation Results

We now demonstrate FARRA algorithm performance for concurrent tracking and classification of multiple targets using a single multimode radar sensor. We consider a scenario consisting of an airborne radar platform and three airborne targets, as illustrated in Figure 10.3. The tracking state vector $\boldsymbol{y}_{n,k}$ is a ten-dimensional vector

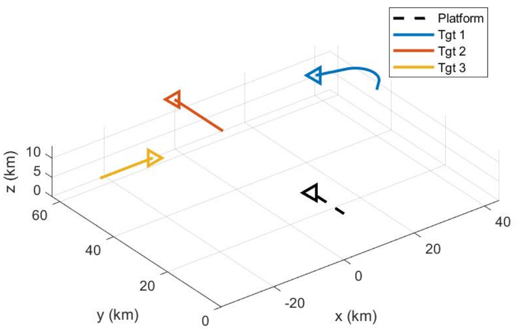


Figure 10.3: Airborne Radar Simulation Scenario

consisting of the nine-dimensional (x,y,z) position, velocity, and acceleration, and the SNR in decibels, which we denote as $s_{n,k} = 10\log_{10}\zeta_{n,k}$, where $\zeta_{n,k}$ is the SNR in linear scale. We use a Singer model [36] for target motion in the tracker. The classification state variable $c_{n,k}$ is assumed to be one of $N_c = 5$ classes, and we assume the transition matrix has diagonal entries $[\boldsymbol{\Upsilon}_n]_{ii} = 0.95$ and off-diagonal entries $[\boldsymbol{\Upsilon}_n]_{ij} = 0.0125$.

We assume the RRA frame is 100ms and the radar sensor must allocate resources to a surveillance task for detecting new targets and to tracking and classification tasks for each of the known targets. We assume 90ms are used for surveillance (search) dwells and the remaining 10ms are for the tracking and classification. In this study, we focus on the tracking and classification tasks and consider the surveillance task only by allocating it a fixed amount of the RRA frame time, thus restricting the time available for the tracking and classification tasks.

The radar system parameters that characterize task performance include the center frequency (f_c), azimuth beamwidth ($\Delta\phi$), elevation beamwidth ($\Delta\theta$), pulse bandwidth (B_p), pulse repetition frequency (f_p), and number of pulses (N_p), as well as the speed of light (c).

The radar can adaptively select between a tracking mode and a classification mode. When in tracking mode, it can choose a pulse repetition frequency (PRF), bandwidth, and pulse count from a list of possibilities. When in classification mode, the radar can select among modes which trade classification accuracy with timeline. The available waveforms, their parameters, and dwell times are listed in Table 10.1. Also included is the "do nothing" waveform #0.

Table 10.1 Waveform Parameters and Dwell Times

Waveform	B_p (MHz)	f_p (kHZ)	N_p	T (ms)
0		N/A		0.0
1,2,3	1,5,10	20	1	0.05
4,5,6	1,5,10	10	1	0.1
7,8,9	1,5,10	20	10	0.5
10,11,12	1,5,10	10	10	1.0
13,14,15	1,5,10	20	20	1.0
16,17,18	1,5,10	10	20	2.0
19,20,21	1,5,10	20	50	2.5
22,23,24	1,5,10	10	50	5.0

Waveform	p_{cc}	T (ms)
25	.3	1.0
26	.6	2.5
27	.75	5.0

In this simulation, the fixed radar parameters are $f_c = 3\text{GHz}$, $\Delta\phi = 2^\circ$, $\Delta\theta = 6^\circ$, $c = 3\times10^8\text{m/s}$, and $P_F = 10^{-6}$.

The FARRA algorithm may elect to measure each target during the dwell or any subset of the targets as long as the total measurement time fits into the allocated time budget. For each target, the sensor may select from the following options:

- **Do nothing** Choose waveform #0. This takes zero time and generates zero utility. It frees up the timeline to dedicate extra dwell time to other targets.
- **Perform a track dwell**. Choose from waveforms #1 – 24. This takes variable time given by N_p/f_p and provides variable utility depending on the waveform parameters.
- **Perform a classification dwell**. Choose from waveforms #25 – 27. This takes variable time and provides variable utility.

The tracking measurement process results in detection-level data of target range (R), range-rate ($\dot{R}$), azimuth angle (ϕ), elevation angle (θ), and SNR in decibels (s). Let $B_{p;n,k}$, $f_{p;n,k}$, and $N_{p;n,k}$ denote the bandwidth, PRF, and number of pulses corresponding to the selected waveform for the nth target. For a given probability of false alarm, P_F, tracking measurements are made with a probability of detection [37]:

$$P_D(\zeta_{n,k};\boldsymbol{\theta}_k) = Q_{MAR}\left(\sqrt{2N_{p;n,k}\zeta_{n,k}}, \sqrt{-2\ln P_F}\right), \tag{10.70}$$

where $Q_{MAR}(a,b)$ is the Marcum Q-function. The estimation covariance matrix is a diagonal matrix whose components are [38, 39]:

$$\begin{aligned}
\left[\boldsymbol{R}_{n,k}(\boldsymbol{\theta}_k)\right]_R &= \left[2N_{p;n,k}\zeta_{n,k}\left(\frac{2}{c}\right)^2\left(3B_{p;n,k}\right)^2\right]^{-1} && (10.71)\\
\left[\boldsymbol{R}_{n,k}(\boldsymbol{\theta}_k)\right]_{\dot{R}} &= \left[2N_{p;n,k}\zeta_{n,k}\left(\frac{4\pi f_c}{c}\right)^2\left(\frac{1}{12B_{p;n,k}^2}+\frac{(N_{p;n,k}^2-1)}{12f_{p;n,k}^2}\right)\right]^{-1} && (10.72)\\
\left[\boldsymbol{R}_{n,k}(\boldsymbol{\theta}_k)\right]_\phi &= \left[2N_{p;n,k}\zeta_{n,k}\left(\frac{1.782\pi}{\Delta\phi}\right)^2\right]^{-1} && (10.73)\\
\left[\boldsymbol{R}_{n,k}(\boldsymbol{\theta}_k)\right]_\theta &= \left[2N_{p;n,k}\zeta_{n,k}\left(\frac{1.782\pi}{\Delta\theta}\right)^2\right]^{-1} && (10.74)\\
\left[\boldsymbol{R}_{n,k}(\boldsymbol{\theta}_k)\right]_s &= \left(\frac{10}{\ln(10)}\right)^2. && (10.75)
\end{aligned}$$

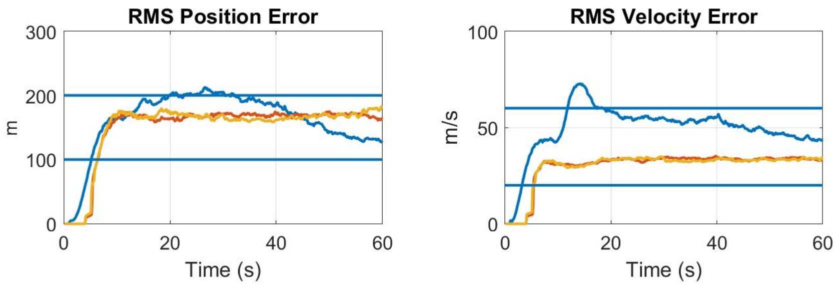
The classification measurement process returns a discrete classification call with probability of correct classification $p_{cc;n,k}$ corresponding to the selected waveform for the nth target. The likelihood matrix has the form

$$[\boldsymbol{L}_n]_{ij}(\boldsymbol{\theta}_k) = \begin{cases} p_{cc;n,k} & i = j \\ \dfrac{1-p_{cc;n,k}}{4} & i \neq j. \end{cases} \tag{10.76}$$

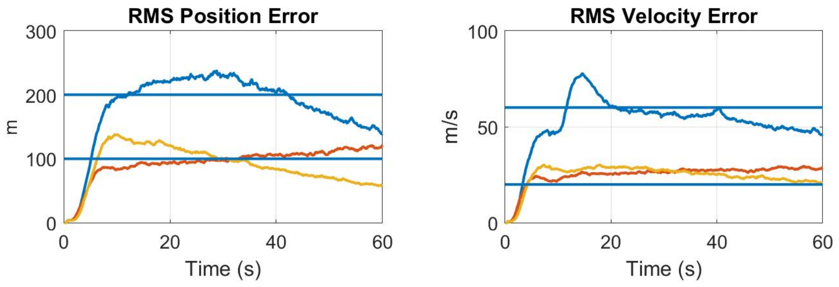
The predicted utility of a sensing action is scored using either the task-driven metric in (10.88) or the information-driven metric in (10.67). For the QoS metric, we set $\bar{G}_1^R = 100$m, $\bar{G}_1^V = 20$m/s, $\bar{G}_1^C$=1.2, $\bar{G}_2^R = \bar{G}_3^R = 200$m, $\bar{G}_2^V = \bar{G}_3^V = 60$m/s, and $\bar{G}_2^C = \bar{G}_3^C$=0.8. The tasks are equally weighted. The objective is then maximized subject to the timeline constraint.

The simulation is repeated for 1000 Monte Carlo trials for each method. The trials have the sensor and target trajectories fixed, but a random realization of the measurements is drawn anew each time. This, in turn, affects the adaptive resource allocation calculations leading to different allocations and performance each time.

Figure 10.4 shows the positon and velocity RMSEs and Figure 10.5 shows the classification entropy and probability of corrrect classification for each method. Also shown are the performance goals used in the task-driven method. These are not used in the information-driven method, but are shown for reference. Figure 10.6 shows the MI for each method. This is not used in the task-driven method, but is shown for reference. Figure 10.7 shows how the resource allocation algorithm selected to use the sensors over time by looking at what fraction of the 10ms frame is used for each target at each time.



(a) Task-Driven Method

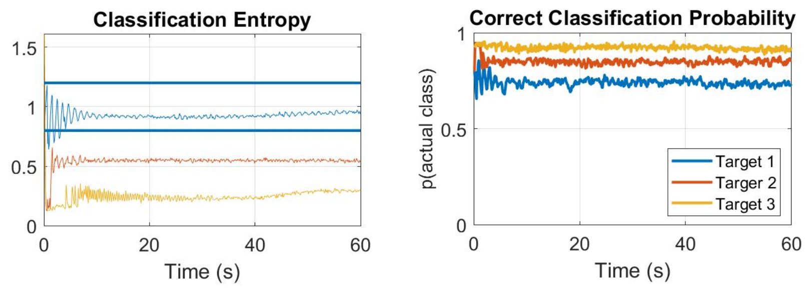


(b) Information-Driven Method

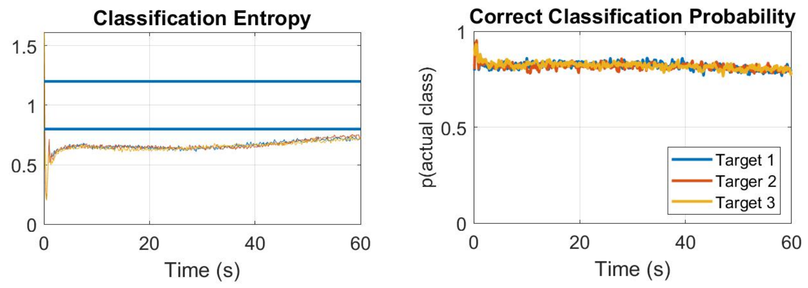
Figure 10.4: Tracking RMSE

This example illustrates that the task-driven and information-driven FARRA algorithms produce similar RMSEs, probability of correct classification, and classification entropy for the three targets. In the task-driven method, Targets 2 and 3 always meet their performance goals, while Target 1 is only able to meet its classification goal. In the information-driven method, where performance goals are not considered, Target 1 RMSE values are slightly higher and Target 2 and 3 RMSE values are slightly lower. The classification entropies are essentially the same for all three targets, with a lower value for Target 1 and higher values for Targets 2 and 3 as compared to the task-driven method. The information-driven method maximizes the total mutual information and does so by making the mutual information approximately the same for each target. The task-driven method does not consider mutual information and achieves a slightly higher value for Target 1 but lower values (on average) for Targets 2 and 3.

The methods the scheduling algorithms deploy to reach the roughly equal tracking and classification performance are different. Broadly speaking, we find that both approaches interleave tracking and classification, with more classification dwells during the first portion of the simulation to maintain track accuracy but also to learn about the target class. After that, classification dwells are taken periodically to main

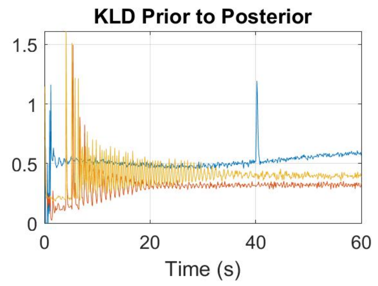


(a) Task-Driven Method

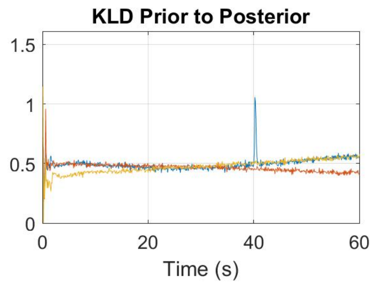


(b) Information-Driven Method

Figure 10.5: Classification Entropy and Probability of Correct Classification



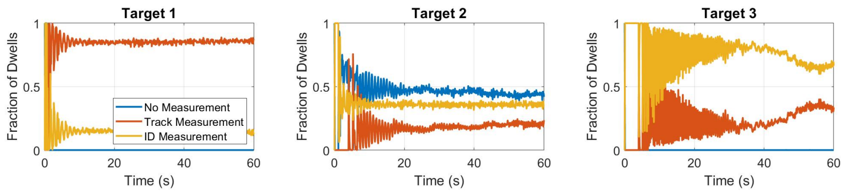
(a) Task-Driven Method



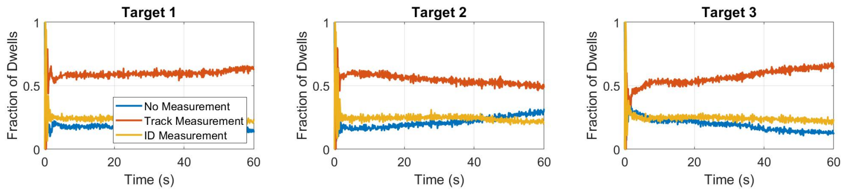
(b) Information-Driven Method

Figure 10.6: Mutual Information

classification performance. For tracking, the task-driven method typically measures one target at 5ms and the other two at 2.5ms. In contrast, the information-driven approach prefers to make 5ms dwells. It does this by typically measuring two targets at 5ms and skipping one target. This generates larger P_D dwells for two targets at the expense of not measuring a third. In the task-driven method, the algorithm is not able to meet the RMSE performance goals for Target 1, but tries very hard by taking tracking measurements about 85% of the time and classification measurements about



(a) Task-Driven Method



(b) Information-Driven Method

Figure 10.7: FARRA Selection of Waveforms

15% of the time. For Targets 2 and 3, the RMSE goals are met easily, and less time is spent on tracking measurements. In the information-driven method, the allocation of tracking, classification, and no measurement dwells is roughly the same for all three targets at approximately 60%, 25%, and 20%, respectively.

10.6 Experimental Results

In this section, we demonstrate FARRA algorithm performance for concurrent tracking and classification of a single target using a single radar sensor in the CREW testbed [29]. The scenario is illustrated Figure 10.8. As the human target moves back and forth in the laboratory, the tracking task estimates the range to the target and its velocity and the classification task separately assigns one of three motion classes: "walking", "jogging/running", and "punching" to the observed target.

The tracking model is the same as in [27]. The state vector $\boldsymbol{y}_k$ is a three-dimensional vector consisting of the target's range (R), velocity (V), and the pulse-integrated SNR in linear scale ($S = N_p\zeta$). We use a nearly constant velocity target motion model in the tracker with an empirically determined process noise covariance matrix [27]. The classification state variable is assumed to be one of $N_c = 3$ classes and the transition matrix has diagonal entries $[\boldsymbol{\Upsilon}_n]_{ii} = 0.99$ and off-diagonal entries $[\boldsymbol{\Upsilon}_n]_{ij} = 0.005$.

The radar sensor must allocate resources to a tracking task and a classification task. During each dwell, the CREW radar transmits linear frequency modulation (LFM) waveform pulse, where the available waveforms consist of different combinations of allowable LFM bandwidths, pulse widths, PRFs, and number of pulses. The CREW can set these parameters with very fine precision, thus the number of

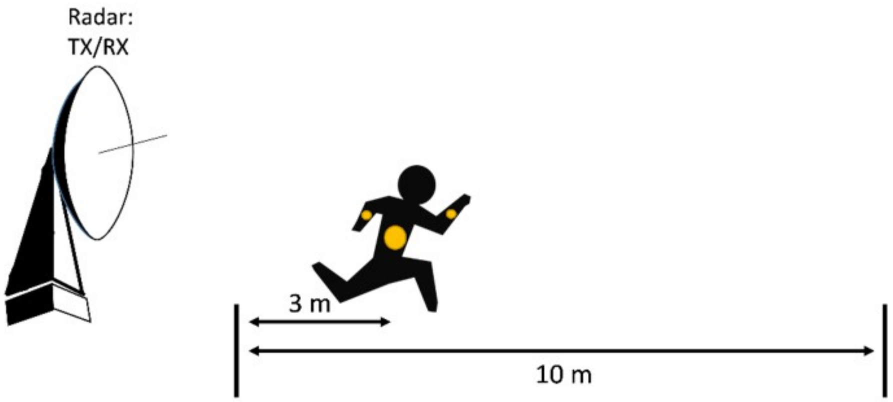


Figure 10.8: CREW Scenario

available waveforms is very large and we don't enumerate all the possibilities, but define the sensor parameter vector in terms of the four adjustable parameters, i.e.,

$$\boldsymbol{\theta}_k = \begin{bmatrix} B_{p;k} & \tau_{p;k} & f_{p;k} & N_{p;k} \end{bmatrix}^T . \tag{10.77}$$

The tracking task processes data from every dwell and tracking information updates are always performed regardless of sensor parameter settings used to collect data. The range of values of the four adjustable parameters is given in Table 10.2. In this experiment we limited each parameter to 10 values. The CREW can also adjust its transmit power (P_t) on each dwell, however in this example we hold it fixed. The fixed transmit power and center frequency are also shown in Table 10.2.

Table 10.2 CREW Tracking Waveform Parameters

Parameter	Number of Values	Adaptive Value Range
f_c	1	95.5 GHz
P_t	1	11.5 dBm
B_p	10	300-1000 MHz
τ_p	10	0.5-1.0 μs
f_p	10	4-15 kHz
N_p	10	64-512

The classifier used in the classification task has been developed using training data collected with two fixed sets of sensor parameters, therefore classification can be performed only when the data is collected using these sensor parameter settings, shown in Table 10.3. If the current data is collected using other sensor parameters, the data is not provided to the classification task and there is no classification information update.

Table 10.3 CREW Classification Waveform Parameters

Waveform	B_p (GHz)	τ_p (μs)	f_p (kHZ)	N_p
Low	0.3	0.5	4.889	64
High	1.0	0.5	15.0	512

Let $\Delta t(\boldsymbol{\theta}_k)$ denote the measurement update interval. It depends on the length of the CPI and the processing time, t_{proc},

$$\Delta t(\boldsymbol{\theta}_k) = \frac{N_{p;k}}{f_{p;k}} + t_{proc}. \tag{10.78}$$

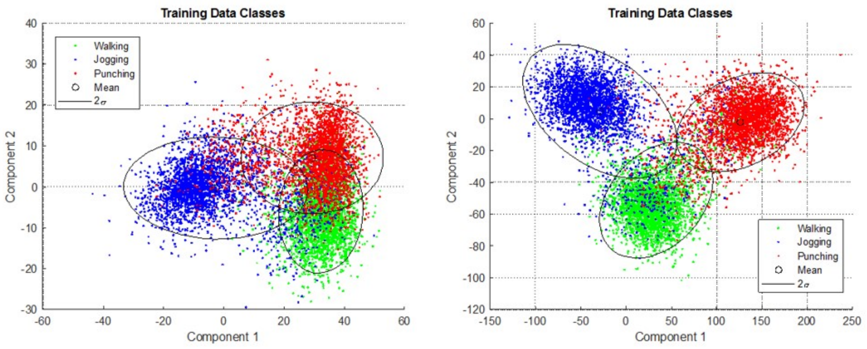
The FARRA algorithm must decide the sensor parameters to use during each dwell. If it chooses parameters to optimize tracking performance, then no data may be provided to the classifier. If it chooses parameters to optimize classification performance, then data is still provided to the tracker, but it may be of less utility. There is no "do nothing" option in this experiment.

For the tracking task, the data is range-Doppler processed and detection-level data of target range (R), Doppler frequency (F), and pulse-integrated SNR (S) are obtained. The transmit power is high enough that the probability of detection is equal to one. The estimation covariance matrix is a diagonal matrix whose components are [27]:

$$\begin{aligned} [\boldsymbol{R}_k(\boldsymbol{\theta}_k)]_R &= \left[C_R S_k B_{p;k}^2\right]^{-1} && (10.79) \\ [\boldsymbol{R}_k(\boldsymbol{\theta}_k)]_F &= \left[C_F S_k \left(\frac{N_{p;k}}{f_{p;k}}\right)^2\right]^{-1} && (10.80) \\ [\boldsymbol{R}_k(\boldsymbol{\theta}_k)]_S &= \left[C_S N_{p;k}\right]^{-1}. && (10.81) \end{aligned}$$

where the constants C_R, C_F, and C_S were determined through empirical data analysis [27].

For the classification task, the data is processed to produce a two-dimensional feature vector. As described in [31], the classification feature vector is obtained by processing the I/Q data from the CREW sensor to produce a range profile and locate the target range bin. A spectrogram is then computed using the short time Fourier transform, and a high-dimensional feature vector is formed from 500 normalized spectral samples. This was repeated many times to obtain a set of training data, and multiple discriminant analysis (MDA) was performed to obtain a matrix for projection of the full 500-dimensional feature vector down to a two-dimensional (2D) feature vector. The projection matrix was stored, and is used to project the high-dimensional data down to the 2D feature vector space. Figure 10.9 shows the 2D feature vectors obtained after MDA and projection to 2D space for two training datasets collected with the two different radar parameter settings. Note that the feature space vectors are unique up to an arbitrary angular rotation in 2D space, thus this has to be recognized and ignored when comparing the datasets. The plots show that the high performance waveform produces much tighter class clusters than the low performance waveform. In particular, the walking (legs moving) and punching (hands moving) classes of the low performance waveform have a large overlap resulting in increased classification uncertainties for these classes.



(a) Low Performance Waveform (b) High Performance Waveform

Figure 10.9: Feature Space Comparison of CREW Measured Target Returns for Low and High Performance Waveforms

The means and covariance matrices of the 2D feature vectors shown in Figure 10.9 are determined from the sample mean and covariance of the clusters. The mean is the center of the cluster and the covariance is represented by the 2σ error ellipse overlaid on the data. These are the values of $\boldsymbol{\mu}_i(\boldsymbol{\theta}_k)$ and $\boldsymbol{\Sigma}_i(\boldsymbol{\theta}_k)$ used in the likelihood function in (10.28).

For this example, we use the multi-objective optimization cost function approach to develop a task-based FARRA objective function [21]. In this approach, cost functions are specified rather than utility functions, and the objective function is minimized. The tracking QoS metrics are the range and velocity RMSEs obtained from the PC-CRLB and the classification QoS metric is the conditional entropy:

$$G_k^R(\boldsymbol{\theta}_k) = \sqrt{\left[\tilde{\boldsymbol{C}}_k^{\uparrow}(\boldsymbol{\theta}_k)\right]_R} \tag{10.82}$$

$$G_k^V(\boldsymbol{\theta}_k) = \sqrt{\left[\tilde{\boldsymbol{C}}_k^{\uparrow}(\boldsymbol{\theta}_k)\right]_V} \tag{10.83}$$

$$G_k^C(\boldsymbol{\theta}_k) = H^{\uparrow}(\boldsymbol{\theta}_k|\boldsymbol{Z}_{k-1};\boldsymbol{\Theta}_{k-1}). \tag{10.84}$$

The QoS requirements are the values that we want the RMSEs and conditonal entropy to be below, denoted as $\bar{G}^R$, $\bar{G}^V$, and $\bar{G}^C$. We then define the position, velocity, and entropy cost functions to be:

$$C_k^R(\boldsymbol{\theta}_k) = \begin{cases} \frac{G_k^R(\boldsymbol{\theta}_k) - \bar{G}^R}{\bar{G}^R} & G_k^R(\boldsymbol{\theta}_k) > \bar{G}^R \\ 0 & G_k^R(\boldsymbol{\theta}_k) \le \bar{G}^R \end{cases} \tag{10.85}$$

$$C_k^V(\boldsymbol{\theta}_k) = \begin{cases} \frac{G_k^V(\boldsymbol{\theta}_k) - \bar{G}^V}{\bar{G}^V} & G_k^V(\boldsymbol{\theta}_k) > \bar{G}^V \\ 0 & G_k^V(\boldsymbol{\theta}_k) \le \bar{G}^V \end{cases} \tag{10.86}$$

$$C_k^C(\boldsymbol{\theta}_k) = \begin{cases} \frac{G_k^C(\boldsymbol{\theta}_k) - \bar{G}^C}{\bar{G}^C} & G_k^C(\boldsymbol{\theta}_k) > \bar{G}^C \\ 0 & G_k^C(\boldsymbol{\theta}_k) \le \bar{G}^C. \end{cases} \tag{10.87}$$

With these cost functions, if the QoS metric is below the required value, the resulting cost is zero and there is neither a penalty nor any additional utility for being below the requirement. The mission processing cost function is obtained by assigning weights to the task cost functions, which we denote as w_R, w_V, and w_C, and computing the weighted sum of the task costs,

$$C_P(\boldsymbol{\theta}_k) = w_R C_k^R(\boldsymbol{\theta}_k) + w_V C_k^V(\boldsymbol{\theta}_k) + w_C C_k^C(\boldsymbol{\theta}_k). \tag{10.88}$$

In this example, there is no hard constraint on the observation time, however, we define a measurement cost function to characterize user preferences for parameter selections. The preferred sensor parameter values are denoted as $\bar{B}_p$, $\bar{\tau}_p$, $\bar{f}_p$, and $\bar{N}_p$ and the measurement cost weights are denoted as w_B, w_τ, w_f, and w_N. The measurement cost function is defined as:

$$C_M(\boldsymbol{\theta}_k) = w_B \left| \frac{B_{p;k} - \bar{B}_p}{\bar{B}_p} \right| + w_\tau \left| \frac{\tau_{p;k} - \bar{\tau}_p}{\bar{\tau}_p} \right| + w_f \left| \frac{f_{p;k} - \bar{f}_p}{\bar{f}_p} \right| + w_N \left| \frac{N_{p;k} - \bar{N}_p}{\bar{N}_p} \right|. \tag{10.89}$$

Finally, the executive cost function is the sum of the measurement and processor cost functions,

$$C_E(\boldsymbol{\theta}_k | \boldsymbol{Z}_{k-1}; \boldsymbol{\Theta}_{k-1}) = C_M(\boldsymbol{\theta}_k) + C_P(\boldsymbol{\theta}_k) \tag{10.90}$$

The next action vector is then determined by minimizing the executive cost function

$$\boldsymbol{\theta}_k = \arg\min_{\boldsymbol{\theta}} C_E(\boldsymbol{\theta} | \boldsymbol{Z}_{k-1}; \boldsymbol{\Theta}_{k-1}). \tag{10.91}$$

The QoS requirements, goal values, and weights for this example and given in Table 10.4. Following [21], we chose the goal values and weights to favor timeline minimization. The time required to update a target track is the sum of the CPI and the processing time, with the CPI being the dominant factor. We set the goal PRF to the highest value and the goal number of pulses to the lowest value. The processing time is a function of the amount of data the radar has to process. The number of fast-time samples collected with each pulse increases with pulse width o small pulse width is preferable for this case. Similarly, decreasing the waveform bandwidth decreases the Nyquist sampling rate, so lower sampling frequencies can be used. This decreases the number of samples per measurement, but only if the sampling frequency is adjustable. This is not a feature of the CREW system, but the bandwidth was still included in the optimization to demonstrate how such a flexible system might benefit. Both pulse width and bandwidth impacted the overall track update time less than the CPI, so they were given weights of one, while PRF and N_p were given weights of 20 and 10, respectively. The PRF weight was double the N_p weight to reflect an additional preference for higher PRFs to avoid Doppler aliasing.

Table 10.4 QoS Requirements, Goal Values, and Weights

Parameter	Requirement/Goal	Weight
B_p	300 MHz	1
τ_p	0.5 μs	1
f_p	15 kHz	20
N_p	64	10
R	0.1 m	1/3
V	0.1 m/s	1/3
C	0.3	1/3

In this example, we also explored incorporating measurement cost and task weighting in the information-based approach. We created cost functions for the tracking and classification tasks, assigned weights to the tasks which we denote as w_T and w_C, and summed to form the processor cost function,

$$\begin{aligned} C_k^T(\boldsymbol{\theta}_k) &= -I_{\mathbf{yz}}(\boldsymbol{\theta}_k|\mathbf{Z}_{k-1};\boldsymbol{\Theta}_{k-1}) & (10.92)\\ C_k^C(\boldsymbol{\theta}_k) &= -I_{c\boldsymbol{\xi};n}(\boldsymbol{\theta}_k|\mathbf{Z}_{k-1};\boldsymbol{\Theta}_{k-1}) & (10.93)\\ C_P(\boldsymbol{\theta}_k) &= w_T C_k^T(\boldsymbol{\theta}_k) + w_C C_k^C(\boldsymbol{\theta}_k). & (10.94)\end{aligned}$$

Note that the costs defined above can be less than zero. We also incorporate the measurement cost function defined in (10.89), this time weighted by w_M,

$$C_E(\boldsymbol{\theta}_k|\mathbf{Z}_{k-1};\boldsymbol{\Theta}_{k-1}) = w_M C_M(\boldsymbol{\theta}_k) + C_P(\boldsymbol{\theta}_k). \qquad (10.95)$$

For this example, we chose $w_T = 1$, $w_C = 6$, and $w_M = 0.1$. With these weights, the measurement the tracking, classification, and measurement costs are roughly the same order of magnitude.

The predicted cost of a sensing action is scored using either the task-based metric in (10.90) or the information-based metric in (10.95). The cost is then minimized. Our computational approach is to use Matlab's 'fmincon' sequential quadratic programming algorithm in the Optimization Toolbox. Despite the discrete set of available parameters in the waveform library, the optimization was solved over a continuous parameter space, and each parameter final solution was rounded up to the nearest available value. The rounding approach results in an overspending of resources, but we preferred this solution because the continuous space optimizations were faster than explicitly solving the discrete parameter problem.

The target in this example is initially punching at a close range, then jogging away from the sensor, then reaches the maximum range and stops and punches for a while, then jogs toward the sensor, then reaches the minimum range and stops and punches for a while.

10.6.1 Information-Based FARRA

Figure 10.10 shows the measurement cost, processor cost, and executive cost vs. time. The tradeoff between processor and measurement cost is evident in the plots. Most of the time, the executive cost is minimized by choosing a tracking waveform with low measurement cost and moderate processor cost. However, when the posterior entropy in the classification task gets too high, the classification cost becomes the dominant factor and the executive cost is minimized by choosing the high performance classification waveform with a high measurement cost and low processor cost. Figure 10.11 shows the sensor parameters vs. time.

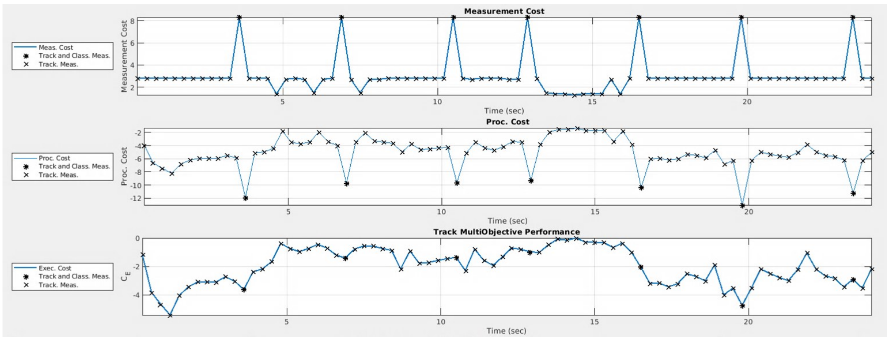


Figure 10.10: Information-Based FARRA Cost Functions vs. Time

Figure 10.12 shows the tracking task performance. The three plots in the first column on the left show the range, velocity, and SNR tracks. The three plots in the second column show Doppler clutter/ambiguity avoidance, velocity RMSE compared to the task-based requirement, and range RMSE compared to the task-based requirement. Because the information-based FARRA algorithm made no attempt to meet the tracking requirements, the velocity RMSE exceeded the requirement most of the time and the range RMSE was well below the requirement most of the time.

Figure 10.13 shows the classification task performance. The left plot shows the posterior entropy compared to the task-based requirement and the right plot shows the posterior class probabilities. The information-based algorithm makes no attempt to meet the requirement, and we see that the entropy gradually increases until a classification measurement is received, then drops significantly because the classifier is able to determine the correct class with high probability with just a single measurement. The correct class probabilities lag the class changes due to the timing of the received measurements.

10.6.2 Task-Based FARRA

In comparison, Figure 10.14 shows the cost functions vs. time for the task-based FARRA algorithm and Figure 10.15 shows the waveform selections. Figure 10.16 shows the tracking task performance and Figure 10.17 shows the classification task performance. Here we see the same tradeoff between processor and measurement

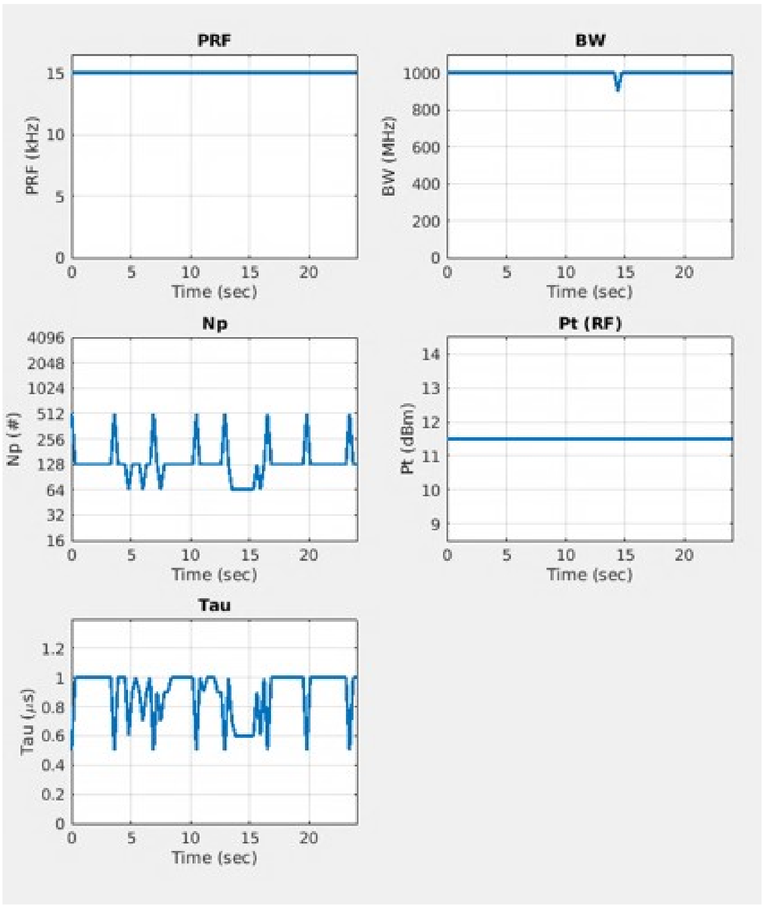


Figure 10.11: Information-Based FARRA Waveform Parameter Selections

cost. Most of the time, the executive cost is minimized by choosing a tracking waveform with low measurement cost and low processor cost. When the posterior entropy in the classification task gets too high, the classification cost becomes the dominant factor and the executive cost is minimized by choosing the high performance classification waveform. In this case, the range and velocity RMSEs are kept very close to the requirements by choosing tracking waveforms most of the time with parameters that vary. Again we see that the entropy gradually increases until a classification measurement is received. In this case, classification measurements are taken more often because the task-based algorithm tries to stay below the entropy requirement. Again, the correct class probabilities lag the class changes due to the timing of the received measurements.

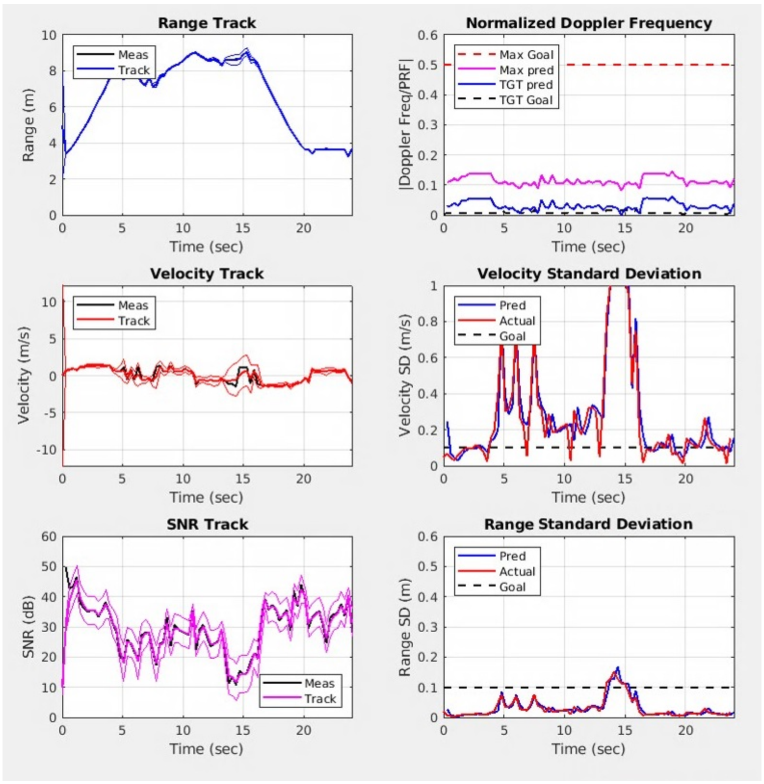


Figure 10.12: Information-Based FARRA Tracking Performance

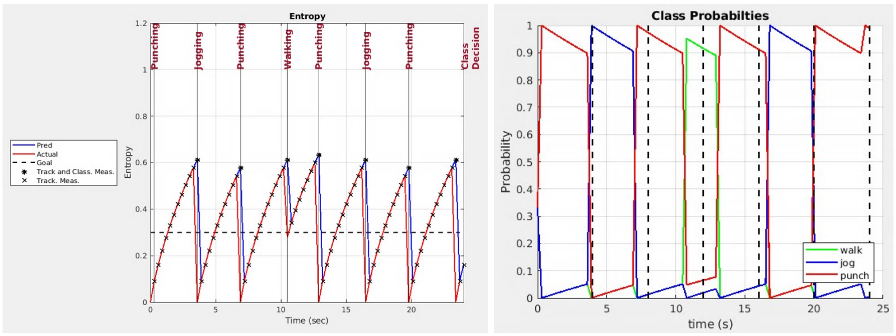


Figure 10.13: Information-Based FARRA Classification Performance

10.7 Conclusion

In this work, we demonstrated FARRA algorithm performance for concurrent tracking and classification of multiple targets using a single radar sensor. The FARRA

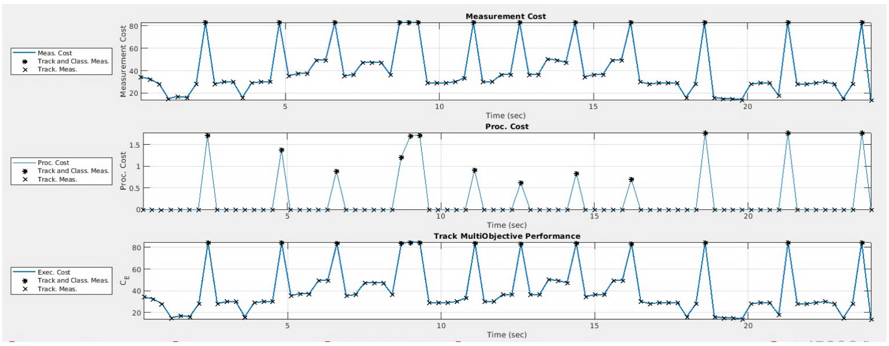


Figure 10.14: Task-Based FARRA Cost Functions vs. Time

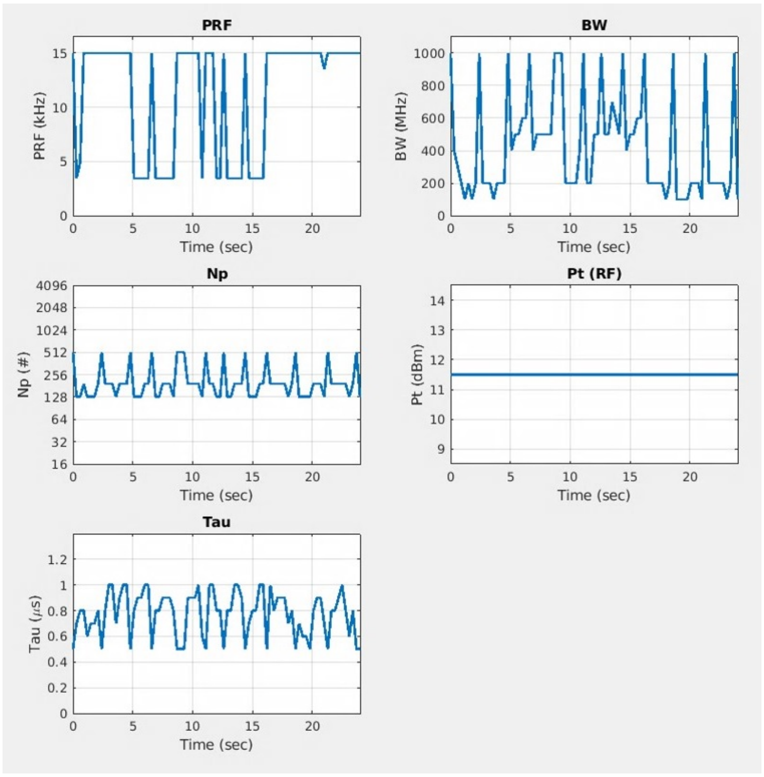


Figure 10.15: Task-Based FARRA Waveform Parameter Selections

approach is based on the perception-action cycle of cognition and includes a perceptual processor that performs multiple radar system tasks and an executive processor that allocates system resources to the tasks to decide the next transmission of the radar on a dwell-by-dwell basis. We used a simulation to model a scenario con-

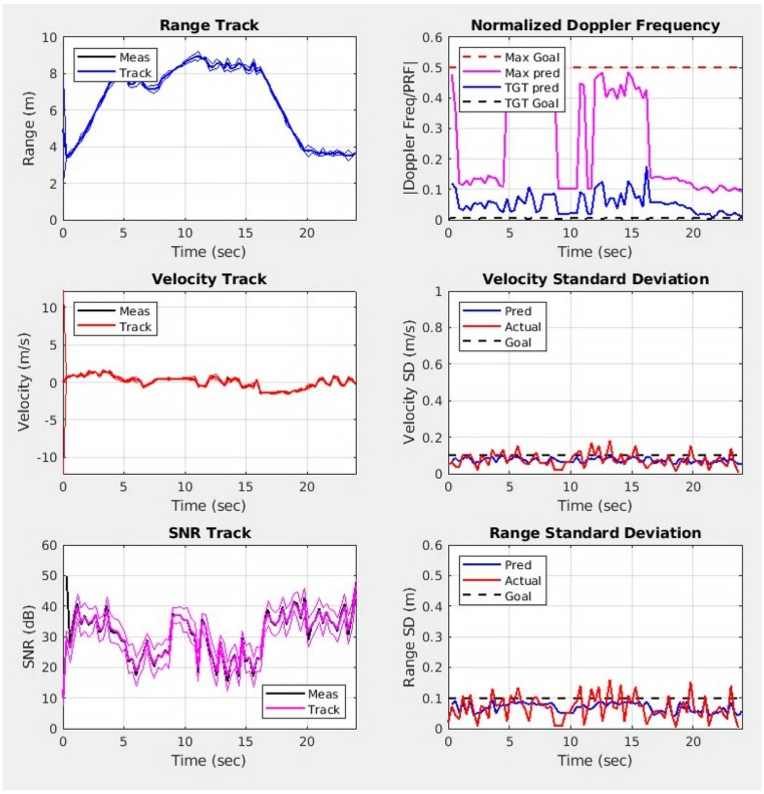


Figure 10.16: Task-Based FARRA Tracking Performance

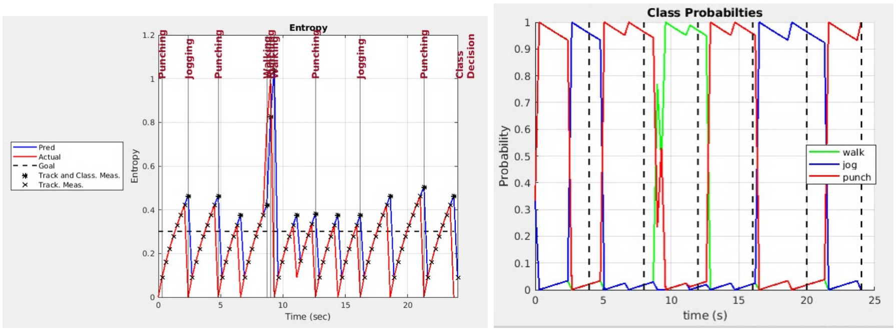


Figure 10.17: Task-Based FARRA Classification Performance

sisting of an airborne radar platform and multiple airborne targets and the CREW testbed to model a scenario consisting of a single moving target and a single stationary sensor. In both cases, we presented examples to demonstrate the application of task-based and information-based FARRA algorithms to simultaneous tracking and classification.

We showed that the task and information-based algorithms were actually based on the same information-theoretic quantities, and the examples showed that the two methods had similar tracking and classification performance but selected different parameter sets to achieve their solutions. Furthermore, the task-based and information-based algorithms had essentially the same computational complexity since their objective functions were based on the same fundamental quantities and they used the same methods for solving the executive processor optimization problem.

References

[1] Haykin S, Xue Y, Setoodeh M. Cognitive radar: step toward bridging the gap between neuroscience and engineering. Proceedings of the IEEE. 2012;100(11):3102–3130.

[2] Haykin S. Cognitive Dynamic Systems: Perception-Action Cycle, Radar and Radio. Cambridge University Press; 2012.

[3] Guerci JR. Cognitive Radar: The Knowledge-Aided Fully Adaptive Approach. Artech House; 2010.

[4] Greco M, Gini F, Stinco P, et al. Cognitive radars: on the road to reality: progress thus far and possibilities for the future. IEEE Signal Processing Magazine. 2018;35(4):112–125.

[5] Gurbuz S, Griffiths H, Charlish A, et al. An overview of cognitive radar: past, present, and future. IEEE Aerospace and Electronic Systems Magazine. 2019;34(12):6–18.

[6] Fuster JM. Cortex and Mind: Unifying Cognition. Oxford University Press; 2010.

[7] Hernandez M, Kirubarajan T, Bar-Shalom Y. Multisensor resource deployment using posterior Cramer-Rao bounds. IEEE Transactions on Aerospace and Electronic Systems. 2004;40(2):399–416.

[8] Kreucher C, Kastella K, Hero AO. Multi-target sensor management using alpha-divergence measures. In: Proc. Information Processing in Sensor Networks; 2003. p. 209–222.

[9] Kreucher C, Hero AO, Kastella K, et al. Efficient methods of non-myopic sensor management for multitarget tracking. In: Proc. 43rd IEEE Conference on Decision and Control; 2004. p. 722–727.

[10] Kreucher C, Hero A, Kastella K. A comparison of task driven and information driven sensor management for target tracking. In: Proc. 44th IEEE Conference on Decision and Control; 2005. p. 4004–4009.

[11] Kreucher CM, Hero AO, Kastella KD, et al. Information-based sensor management for simultaneous multitarget tracking and identification. In: Proc. 13th Conference on Adaptive Sensor Array Processing; 2005. .

[12] Kreucher C, Kastella K, Hero AO. Multi-platform information-based sensor management. In: Proc. SPIE; 2005. p. 141–151.