Comparison of Myopic and Lookahead Scan Strategies for Meteorological Radars

Victoria Manfredi, Jim Kurose *

Technical Report 2006-62
December 19, 2006

Department of Computer Science
140 Governors Drive
University of Massachusetts
Amherst, Massachusetts 01003-4601
{vmanfred, kurose}@cs.umass.edu

Abstract

While most meteorological radars, such as the National Weather Service NEXRAD system, are traditionally tasked to always scan 360 degrees, a new generation of small, low-power but agile radars are being developed that can perform sector scanning, targeting sensing resources when and where the user needs are greatest. In this work, we compare four strategies to adapt the scanning of a network of weather radars from one decision epoch to the next. The myopic strategies use past and current environmental state but do not estimate future states when making control decisions. The limited lookahead strategy additionally uses the expected environmental state k decision epochs in the future, as predicted from Kalman filters, in its decision-making. The full lookahead strategy uses all expected future states by casting the problem as a Markov decision process and using reinforcement learning to estimate the optimal scan strategy. We show that the main benefits of using a lookahead strategy are when there are multiple meteorological phenomena in the environment, and when the maximum radius of any phenomenon is sufficiently smaller than the radius of the radars. We also show that there is a trade-off between the average quality with which a phenomenon is scanned and the number of decision epochs before which a phenomenon is rescanned.

Keywords: Sensor tasking and control, Energy and resource management

1 Introduction

Traditionally, meteorological radars, such as the National Weather Service NEXRAD system, are tasked to always scan 360 degrees. The Collaborative Adaptive Sensing of the Atmosphere (CASA) Engineering Research Center [24] is building a new generation of small, low-power but agile radars that can perform sector scanning, targeting sensing resources when and where the user needs are greatest. The ability for a radar to

*This work was supported in part by the National Science Foundation under grants EEC-0313747001 and ECS-0218125. Any opinons, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

scan a subsector, rather than 360 degrees, has several advantages for detecting meteorological phenomena. In particular, given a fixed interval of time, the smaller the sector that is scanned by the radar, the more time that is spent scanning the sector. Consequently, more sensing energy is focused into the sector, resulting in higher-quality measurements until some minimum sector size is reached. Higher-quality measurements can be obtained either by spending more time scanning a single elevation, resulting in better statistical averages; or by scanning multiple elevations rather than only a single elevation in the same amount of time.

Given the ability of a network of radars to perform sector scanning, how should scanning be adapted at each decision epoch? Any scan strategy must consider, for each scan action, both the expected quality with which phenomena would be observed, and the expected number of decision epochs before which phenomena would be first observed or rescanned (since now not all regions are scanned every epoch). Another consideration is whether to optimize myopically only over current and possibly past environmental state, or whether to additionally optimize over expected future states. For instance, suppose a storm cell is about to move into a high-quality multi-doppler region (i.e., the area where multiple radar footprints overlap). By predicting the future location of the storm cell, a lookahead strategy can anticipate this event and have all radars focused on the storm cell when it enters the multi-doppler region, rather than expending resources (with little "reward") to scan the cell just before entering this region.

In this work we examine four methods for adapting the radar scan strategy. The methods differ in the information they use to select a scan configuration at a particular decision epoch. The sit-and-spin strategy of always scanning 360 degrees is independent of any external information. The myopic strategy uses past and current environmental state but does not estimate future states when making control decisions. The limited lookahead strategy additionally uses the expected environmental state k decision epochs in the future in its decision-making. Finally, the full lookahead strategy has an infinite horizon: it uses all expected future states by casting the problem as a Markov decision process and using reinforcement learning to estimate the optimal scan strategy. All strategies, excluding the sit-and-spin, work by optimizing the overall "quality" (a term we will define precisely shortly) of the sensed information about phenomena in the environment, while restricting or penalizing long inter-scan intervals. We show that the main benefits of using a lookahead strategy are when there are multiple meteorological phenomena in the environment, and when the maximum radius of any phenomenon is sufficiently smaller than the radius of the radars. We also show that there is a trade-off between the average quality with which a phenomenon is scanned and the number of decision epochs before which a phenomenon is rescanned.

Related work by [6] and [20] shows that lookahead radar scan strategies for evasive target-tracking outperform myopic strategies. Although [6] considers the problem of tracking of ground targets using agile radars on airplanes, we consider the problem of tracking meteorological phenomena using ground radars. [20] considers 1-step and 2-step look-ahead strategies; an infinite horizon strategy is not considered.

The rest of this paper is organized as follows. Section 2 defines the radar scheduling problem. Section 3 describes the myopic and lookahead radar scan strategies we consider. Section 4 discusses the Markov-decision-process-based scan strategy in more depth and presents the associated learning algorithm. Section 5 describes our evaluation framework and presents results comparing the full lookahead scan strategy approach with sit-and-spin, myopic, and limited lookahead strategies. Section 6 reviews related work on scheduling and resource allocation in radar and sensor networks. Finally, Section 7 summarizes this work and outlines future work.

2 Radar Scheduling Problem

In this section we formulate the multi-radar meteorological scheduling problem as follows. We have a set of radars, with fixed locations and possibly overlapping footprints. Each radar has a set of scan actions from which it chooses. In the simplest case, a radar action determines the size of the sector to scan, the start angle, the end angle, and the angle of elevation. We will not consider elevation angles. In this work, we assume that a radar can scan $90°, 180°, 270°$, and $360°$ sectors, where the starting scan angle must be either

$0°, 90°, 180°$, or $270°$. This gives 13 actions per radar. Our goal is to determine which scan actions to use, and when to use them.

Meteorological radar sensing characteristics are such that the smaller the sector that a radar scans (until a minimum sector size is reached), the higher the quality of the data collected, and thus, the more likely it is that phenomena located within the sector are correctly identified [2]. An effective scanning strategy must balance scanning small sectors, to ensure that phenomena are correctly identified, with scanning a variety of sectors, to ensure that no phenomena are missed.

We will evaluate the performance of the different scan strategies based on inter-scan time, quality, and cost. Inter-scan time is the number of decision epochs before a phenomenon is either first observed or rescanned; we would like this value to be below some threshold. Quality measures how well a radar observes a phenomenon, with quality depending on the amount of time a radar spends sampling a voxel in space, the degree to which a meteorological phenomena is scanned in its (spatial) entirety, and the number of radars observing a phenomenon; higher quality scans are better. Cost is a meta-metric that combines inter-scan time and quality, and that additionally considers whether a phenomenon was never scanned. The radar scheduling problem is that of dynamically choosing the scan strategy of the radars over time to optimize quality while minimizing inter-scan time.

3 Scan Strategies

This section defines the quality functions that we use. We define a *radar configuration* to be the start and end angles of the sector to be scanned by an individual radar for a fixed interval of time. We define a *scan action* to be a set of radar configurations. We define a *scan strategy* to be an algorithm for choosing scan actions. In Section 3.1 we define the quality function associated with different radar configurations and in Section 3.2 we define the quality functions associated with different scan strategies.

3.1 Quality Function

The quality function associated with a given scan action has two components. There is a quality component U_p associated with scanning a particular phenomenon p. There is also a quality component U_s associated with scanning a sector, which is independent of any phenomena in that sector. Let s_r be the radar configuration for a single radar r and let S_r be the scan action under consideration. From [11], we compute the quality $U_p(p, S_r)$ of scanning a phenomenon p using scan action S_r with the following equations,

$$\begin{aligned} U_p(p, s_r) &= F_c\left(c(p, s_r)\right) \times \left[\beta F_d\left(d(r, p)\right) + (1-\beta) F_w\left(\frac{w(s_r)}{360}\right)\right] \\ U_p(p, S_r) &= \max_{s_r \in S_r}\left[U_p(p, s_r)\right] \end{aligned} \tag{1}$$

where

$$\begin{aligned} w(s_r) &= \text{size of sector } s_r \text{ scanned by } r \\ a(r, p) &= \text{minimal angle that would allow } r \text{ to cover } p \\ c(p, s_r) &= \frac{w(s_r)}{a(r, p)} = \text{coverage of } p \text{ by } r \text{ scanning } s_r \\ h(r, p) &= \text{distance from } r \text{ to geometric center of } p \\ h_{max}(r) &= \text{range of radar } r \\ d(r, p) &= \frac{h(r, p)}{h_{max}(r)} = \text{normalized distance from } r \text{ to } p \\ \beta &= \text{tunable parameter} \end{aligned}$$

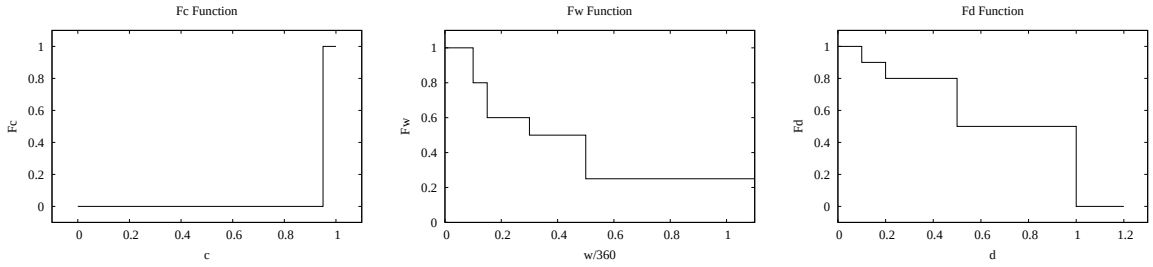


Figure 1: Step functions used by the U_p and U_s quality functions, from [9]

$U_p(p, S_r)$ is the maximum quality obtained for scanning phenomenon p over all possible radars and their associated radar configurations s_r. $U_p(p, s_r)$ is the quality obtained for scanning phenomenon p using a specific radar r and radar configuration s_r. The functions $F_c(\cdot)$, $F_w(\cdot)$, and $F_d(\cdot)$ from [11] are plotted in Figure 1; they are derived from discussions with radar meteorologists. F_c captures the effect on quality due to the percentage of the phenomenon covered; to usefully scan a phenomenon, at least 95% of the phenomenon must be scanned. F_w captures the effect of radar rotation speed on quality; as rotation speed is reduced, quality increases, but if scanning is done too slowly, scan quality begins to degrade. F_d captures the effects of the distance from the radar to the geometrical center of the phenomenon on quality; the further away the radar center is from the phenomenon being scanned, the more degraded will be the scan quality due to attenuation. Note that as a consequence of the F_w function, the quality function $U_p(p, s_r)$ outputs the same quality for scan angles of 181 to 360 degrees.

We compute the quality $U_s(r_i, s_r)$ for scanning a subsector i of radar r scanned using configuration s_r with the equation,

$$U_s(r_i, s_r) \quad = \quad F_w\left(\frac{w(s_r)}{360}\right) \tag{2}$$

We have now defined the components U_p and U_s of the quality function. The quality function that is actually optimized, however, depends on the individual scan strategy. We will see how U_p and U_s are used in the quality functions that are optimized by the individual scan strategies in Section 3.2.

We now define how the quality for a scanned phenomenon, U_p, and the quality for a scanned sector, U_s, decays over time. We assume that for those sectors or previously observed phenomena that the radars do not scan at decision epoch t, their utilities are decayed by a fixed amount κ,

$$\begin{aligned} \text{If } U_p \geq \kappa_p \qquad & U_p = U_p - \kappa_p \\ \text{else} \qquad & U_p = 0 \\ \text{If } U_s \geq \kappa_s \qquad & U_s = U_s - \kappa_s \\ \text{else} \qquad & U_s = 0 \end{aligned}$$

Intuitively, a sector scanning strategy is only preferable when the quality functions U_p and U_s allow the quality gained for scanning a sector to be greater than the quality lost for not scanning another sector.

3.2 Scan Strategies

We compare the performance of the following four scan strategies. All strategies except sit-and-spin optimize quality and either restrict or penalize inter-scan intervals greater than a threshold. The strategies differ primarily in whether they optimize quality over only the current environmental state or whether they additionally optimize over future expected states.

- *Sit-and-spin strategy.* All radars always scan 360°.
- *Myopic strategy.* We compute the myopic quality, represented by $U_m(S_r|T_r)$, for different sets of radar configurations S_r with the following equation based on the U_p quality function defined in Section 3.1.

$$U_m(S_r|T_r) = \sum_p U_p(p, S_r|T_r)$$

The optimal set of radar configurations is then given by $S_r^* = \text{argmax}_{S_r} U_m(S_r|T_r)$. To account for the decay of quality for unscanned sectors and phenomena, and to consider the possibility of new phenomena appearing, we restrict S_r to be those scan actions that ensure that every sector has been scanned at least once in the last T_r decision epochs. T_r is a tunable parameter whose purpose is to satisfy the meteorologists' requests, as specified in [11], that all sectors be scanned, for instance by a 360° scan, at most every 5 minutes.

- *Limited "lookahead" strategy.* We examine two limited lookahead scan strategies: a 1-step look-ahead and a 2-step look-ahead. Although we do not have an exact model of the dynamics of different phenomena, to perform the look-ahead we estimate the future attributes of each phenomenon using a separate Kalman filter [22]. For each Kalman filter, we consider the true state $\mathbf{x}$ to be a vector comprising the (x, y) location and velocity of the phenomenon, and the measurement $\mathbf{y}$ to be a vector comprising only the (x, y) location. The Kalman filter assumes that the state at time t is a linear function of the state at time $t-1$ plus some Gaussian noise, and that the measurement at time t is a linear function of the state at time t plus some Gaussian noise. In particular,

$$\begin{aligned} \mathbf{x}_t &= \mathbf{A}\mathbf{x}_{t-1} + N[\mathbf{0}, \mathbf{Q}] \\ \mathbf{y}_t &= \mathbf{B}\mathbf{x}_t + N[\mathbf{0}, \mathbf{R}] \end{aligned}$$

We initialize each Kalman filter with the following parameters. The $\mathbf{A}$ matrix reflects the knowledge that storm cells typically move to the north-east. The $\mathbf{B}$ matrix, which when multiplied with $\mathbf{x}_t$ returns $\mathbf{x}_t$, reflects our assumption that the observed state $\mathbf{y}_t$ is directly the true state $\mathbf{x}_t$ plus some Gaussian noise. The $\mathbf{Q}$ matrix reflects our assumption that there is little noise in the true state dynamics. Finally, for any scan action, the size of the scan angle and the distance from the radar to the phenomenon affects how well the phenomenon is observed. Thus, the measurement error covariance matrix $\mathbf{R}$ is a function of the quality U_p with which phenomenon p was scanned at time t. We discuss how to compute the σ_t's in Section 4.

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \mathbf{Q} = \begin{bmatrix} .0001 & 0 & 0 & 0 \\ 0 & .0001 & 0 & 0 \\ 0 & 0 & .0001 & 0 \\ 0 & 0 & 0 & .0001 \end{bmatrix}, \mathbf{R} = \begin{bmatrix} \sigma_t & 0 \\ 0 & \sigma_t \end{bmatrix}$$

We use the first location measurement of a storm cell $\mathbf{y}_0$, augmented with the observed velocity, as the the initial state $\mathbf{x}_0$. For the initial covariance $\mathbf{P}_0$ we use $.0001 * I$, assuming that our estimate of $\mathbf{x}_0$ has little noise.

Building on the myopic quality equation, we compute the k-step look-ahead quality for different sets of radar configurations S_r with the following equation,

$$U_K(S_{r,1}|T_r) = \sum_{k=1}^{K} \phi^{k-1} \sum_{i=1}^{N_p} U_p(p_{i,k}, S_{r,k}|T_r)$$

where N_p is the number of phenomena in the environment in the current decision epoch, $p_{i,0}$ is the current set of observed attributes for phenomenon i, $p_{i,k}$ is the k-step set of predicted attributes for phenomenon i, $S_{r,k}$ is the set of radar configurations for the kth decision epoch in the future, and ϕ is a tunable discount factor between 0 and 1. The optimal set of radar configurations is then $S_{r,1}^* = \text{argmax}_{S_{r,1}} U_K(S_{r,1}|T_r)$. As with the myopic scan strategy, we again restrict S_r to be those scan actions that ensure that every sector has been scanned at least once in the last T_r decision epochs.

- *Full "lookahead" strategy.* We formulate the radar scheduling problem as a Markov decision process and use reinforcement learning as a solution technique to solve the Markov decision process and obtain a lookahead scan strategy. We discuss this strategy in depth in the next section.

We now define the first cost function C_1 that we will use to evaluate the performance of the different scan strategies; we will define a second cost function C_2 in Section 3.2. C_1 is a function of the error between the true state and the observed state and whether all storms have been observed. More precisely,

$$C_1 = \sum_{i=1}^{N_p^o} \sum_{j=1}^{N_d} |d_{ij}^o - d_{ij}| + (N_p - N_p^o) P_m \quad (3)$$

where N_p^o is the observed number of storms, N_d is the number of attributes per storm, d_{ij}^o is the observed value of attribute j of storm i, d_{ij} is the true value of attribute j of storm i, N_p is the true number of storms, and P_m is the penalty for missing a storm. The quality with which a storm is observed will determine how far the observed values of the storm's attributes are from the true values.

4 Markov Decision Processes

In this section we present our formulation of the meteorological radar scheduling problem as a Markov decision process (MDP) and discuss how we use reinforcement learning as a solution technique to solve the MDP.

4.1 MDP Formulation

A Markov decision process (MDP) is a stochastic control process given by the tuple $< S, A, T, C >$ defined as follows. S is the set of states. States are assumed to be perfectly observed and future states are assumed independent of past states given the present state. A is the set of actions. $T : S \times A \times S \rightarrow [0, 1]$ is the transition function, which defines how the state evolves over time. $T(s, a, s')$ is the probability of transitioning from state s to state s' under action a. Finally, $C : S \times A \times S \rightarrow \mathcal{R}$: is the cost function, which encodes the goals of the system. $C(s, a, s')$ is the cost of taking action a in state s and transitioning to state s'.

We formulate the meteorological radar scheduling problem as an MDP as follows. Although there are four types of meteorological phenomena that are of interest in the CASA project (storm cells, areas of significant reflectivity or velocity, and rotations), in this work we only consider storm cells. Thus we have that,

- S is the observed state of the environment. We define the state to be a function of the observed number of storms; the observed x, y velocity of each storm; and the observed dimensions of each storm cell given by x, y center of mass and radius. To model the uncertainty in the environment, we additionally define as part of the state quality variables u_p and u_s based on the U_p and U_s quality functions defined in Equations (1) and (2) in Section 3.1. In particular, u_p is the quality $U_p(\cdot)$ with which each storm cell was observed, and u_s is the current quality $U_s(\cdot)$ of each 90 degree subsector, starting at 0, 90, 180, or 270 degrees.
- A is the set of actions available to the radars. This is the set of radar configurations for a given decision epoch. We restrict each radar to scanning subsectors that are a multiple of 90 degrees, starting at 0, 90, 180, or 270 degrees, as described in Section 2. Thus, with N radars there are 13^N possible actions at each decision epoch.
- The transition function $T(S \times A \times S) \rightarrow [0, 1]$ encodes the *observed* environment dynamics, specifically the appearance, disappearance, and movement of storm cells and their associated attributes. This mapping may be probabilistic. We note that for radar scheduling, the next state really is a function

of not just the current state but also the action executed in the current state. For instance, if a radar scans 180 degrees rather than 360 degrees, then any new meteorological phenomena that appear in the unscanned areas will not be observed. Thus, the new phenomena that will be observed will depend on the scanning action of the radar.

- The cost function $C_2(S, A, S) \rightarrow \mathcal{R}$ encodes the goals of the radar sensing network. We define C_2 as equal to C_1 plus a penalty term for not rescanning a storm within T_r decision epochs. More precisely,

$$C_2 = C_1 + \sum_{i=1}^{N_p} I(t_i) P_r \tag{4}$$

where t_i is the number of decision epochs since storm i was last scanned, P_r is the penalty for not scanning a storm at least once within T_r decision epochs, and $I(t_i)$ is an indicator function that equals 1 when $t_i \geq T_r$.

While a POMDP (partially observable Markov decision process) could be used to model the environmental uncertainty, due to the computational cost of exactly or approximately solving a POMDP with a large state space [9, 13], we chose to formulate the radar scheduling problem as an MDP with quality (or uncertainty) variables as in an augmented MDP [7]. We discuss the implications of true versus observed state further in Section 5.1 when we describe the simulation environment.

4.2 Learning Algorithm

Solving the MDP defined in Section 4.1 for its optimal policy gives a full lookahead scan strategy for the radar scheduling problem. As in [18], define a policy π to be a mapping from states to distributions over actions, specifying the probability with which to execute each action in each state. Given the tuple $< S, A, T, C >$, the optimal policy π^* can be found by using dynamic programming to solve the Bellman equations,

$$Q^\pi(s, a) = \sum_{s'} T(s, a, s') \left[C(s, a, s') + \gamma \min_{a'} Q^\pi(s', a') \right]$$

where s and a are the current state and action respectively, s' and a' are the next state and action, and γ is a discount factor weighing the current received cost against expected future costs. Let us define the value of a state to be the discounted sum of costs received when starting in some state, executing an action, receiving some cost, transitioning to the next state and so on: i.e., $c_0 + \gamma^2 c_1 + \gamma^3 c_3 + \ldots$ where c_i is the cost received at decision epoch i. Then $Q^\pi(s, a)$ represents the discounted expected value that would be achieved by following policy π after taking action a in state s. Note that each possible policy π can give rise to a different action-value function $Q^\pi(s, a)$. By using $Q^\pi(s, a)$ to evaluate different policies, the optimal policy π^* can be found.

In situations where the cost or transition functions are not available, but there is access to an emulator or environment in which actions can be executed and subsequent states and costs observed, reinforcement learning can be used to identify the optimal policy. Rather than solving the Bellman equations to identify the best action to execute in each state, reinforcement learning methods instead seek to estimate $Q^{\pi^*}(s, a)$ by keeping track of the actual sequence of costs received. Since we do not have access to a model that would permit us to obtain a transition function, we use reinforcement learning to solve the MDP formulation of the radar scheduling problem.

Due to the continuous-valued state variables, such as a storm's location, maintaining a table-lookup representation of the action-value function is infeasible. Instead we approximate $Q(s, a)$ as a linear combination of basis functions. By appropriately adjusting a set of weights, the reinforcement learning algorithm learns how best to linearly combine the basis functions and thus obtain the action-value function.

To obtain the basis functions used to approximate the action-value function, we use tile coding [5, 16, 17]. Tile coding works by partitioning the state space into a set of tiles. For example, suppose our state space

Table 1: Linear Sarsa(λ) reinforcement learning algorithm. Adapted from [8][9][10].

1 Initialization:
2 $F \leftarrow$ set of all features
3 $A \leftarrow$ set of all actions
4 $w_{f,b} = 0,\ e_{f,b} = 0, \forall f \in F, \forall b \in A$
5 $s =$ initial state
6 $a =$ initial action *(E.g., sit-and-spin)*
7 Repeat until error δ is sufficiently small
8 Update eligibility traces:
9 $F_s \leftarrow$ set of on features for state s
10 $e_{f,b} \leftarrow \lambda e_{f,b}, \forall f \in F, \forall b \in A$
11 $e_{f,a} \leftarrow e_{f,a} + 1, \forall f \in F_s$
12 Environment step:
13 Take action a, observe cost c and next state s'
14 Choose next action:
15 $F_{s'} \leftarrow$ set of on features for state s'
16 $Q_{s',b} \leftarrow \sum_{f \in F_{s'}} w_{f,b}, \forall b \in A$
17 With probability $1 - \epsilon$: $a' \leftarrow \arg\min_b Q_{s',b}$
18 With probability ϵ: $a' \leftarrow$ random action
19 Learn:
20 $\delta = c - Q_{s,a} + \gamma Q_{s',a'}$
21 $w_{f,b} = w_{f,b} + \alpha \delta e_{f,b}, \forall f \in F, \forall b \in A$
22 Update current state and action:
23 $a = a',\ s = s'$

consists only of one state variable, the x-location of the storm cell. Then in the simplest case, we would choose some number of bins into which to partition the values that x-location can take on. This would result in one single-dimensional tiling. If we had multiple variables, we could also tile the cross-product of variables to get multi-dimensional tilings. Tilings allow us to extract features from the state as follows. A given assignment of values to the state variables (i.e., a given state) maps to a unique "on" tile in each tiling. This gives a binary vector for each tiling, with a 1 for the feature (tile) that is on and 0's for the remaining features. In this way we obtain a basis function from each tiling.

In this work, rather than defining tilings over the entire state space, we define a separate set of tilings for each of the state variables. By defining the tilings over a subspace of the state space, we decrease the time needed to learn the value function but we also decrease the accuracy. Using multiple staggered tilings over the same subspace, however, allows for generalization during learning.

We use linear Sarsa(λ) [18] as the reinforcement learning algorithm, shown in Table 1. $Q_{s,a}$ is the action-value for state s and action a. $w_{f,a}$ is the set of weights used to linearly combine the basis functions obtained for features f and action a. α is the learning rate: it represents the rate at which the weights are updated; α should be inversely proportional to the number of tiles used. γ is the discount factor: it represents how much importance is placed on the future versus the present.

Sarsa(λ) estimates the action-value function $Q(s, a)$ by keeping track of the actual sequence of costs received. The heart of the algorithm is lines 19-21 in Table 1. In particular, based on the cost received for taking action a in state s, the error δ is computed and the weights are updated. The intuition here is that there is an old estimate for the value of taking action a in state s, represented by the action-value $Q_{s,a}$. There is also a new estimate given by the immediate cost just received for taking action a in state s plus the expected value of taking action a' in state s' (recall that s' is the next state to which we transition). The error δ is then the difference between these two estimates. Because we are using function approximation, δ is not used

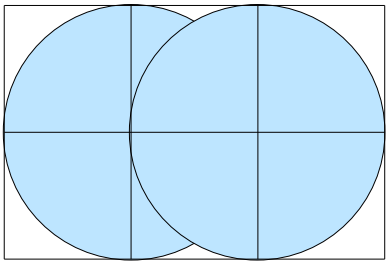


Figure 2: Radar setup for experiments.

to directly update the action-value, rather it used to update the weights, thereby indirectly updating the action-value.

5 Evaluation

In this section we discuss the simulation environment we use to evaluate the different scan strategies and present experimental results obtained using this environment.

5.1 Simulation Environment

We first describe how we simulate the transition function for the MDP in Section 4 to obtain the current state of the environment. We then describe how we parameterize the MDP cost function to evaluate the quality of different radar configurations given the state of the environment. We define each decision epoch to correspond to the 30-second heartbeat interval used in the CASA radar network.

We use two overlapping radars in an $M \times N$ rectangle as in Figure 2, with $M = 90$ km and $N = 60$ km. We consider radars with a 30 km radius, such as the radars in CASA's Oklahoma testbed, and radars with a 10 km radius, such as the radars in the Puerto Rico CASA Student Test Bed [21]. At most a fixed number of storm cells can be present in the environment in any decision epoch. A new storm cell can appear anywhere within the $M \times N$ area. When the (x, y) center of a storm cell is no longer within the range of any radar, the storm cell is removed from the environment.

We derive the maximum storm cell radius from [14], which uses $(2D^2)^{1/2}$ with $D = 2$ km as "the radius from the cell center within which the intensity is greater than e^{-1} of the cell center intensity." Since $(2D^2)^{1/2} = (8)^{1/2} = 2.83$ km, we permit a storm cell's radius to range from 1 to 4 km. To determine the range of storm cell velocities, we use 39 existing storm cell tracks from the National Severe Storms Laboratory courtesy of Kurt Hondl and the WDSS-II software [3]. Each track is a series of $(latitude, longitude)$ coordinates. We first compute the differences in latitude and longitude, and in time, between successive pairs of points. We then fit the differences using Gaussian distributions. Given that the length of a latitude degree at 40° latitude equals 111.04 km and the length of a longitude degree at 40° latitude equals 85.39 km, we obtain, in units of km/hour, that the latitude (or x) velocity has mean 9.1 km/hr and std. dev. of 35.6 km/hr and that the longitude (or y) velocity has mean 16.7 km/hr and std. dev. of 28.8 km/hr. To obtain a storm cell's (x, y) velocity, we then sample the appropriate Gaussian distribution.

To simulate the environment transitions we use a stochastic model of rainfall in which storm cell arrivals are modeled using a spatio-temporal Poisson process, see [14, 1]. To decide whether to add new storm cells during a 30-second interval, we sample a Poisson random variable with rate $\lambda\eta\delta a\delta t$ with $\lambda = 0.075$ storm cells/km^2 and $\eta = 0.006$ storm cells/minute from [14]. From the radar setup we have $\delta a = 90 \cdot 60$ km^2, and from the 30-second decision epoch we have $\delta t = 0.5$ minutes. In this way we obtain the number of new storm cells appearing during each 30-second interval. Given the number of storm cells appearing during the

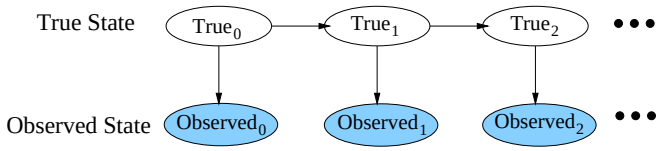


Figure 3: Model of simulation environment. Recall that the observed state of the environment at time t, $Observed_t$, is computed from the true state of the environment at time t, $True_t$ using the equation $Observed_t = True_t + N[0, ((1-u)V^{max}/\rho)^2]$. The true state of the environment at time $t+1$ is computed from the true state of the environment at time t.

interval, we uniformly randomly distribute new storm cells throughout the $M \times N$ region. We then uniformly randomly choose new storm cell attributes from their range of values.

Thus, we can simulate the true state of the environment over time, i.e., the locations of storm cells and their associated attributes, using the methodology above. A scan strategy, however, chooses a set of radar configurations (i.e., a scan action) based on its observed state of the environment. We use a simplified radar model to determine how well the radars observe the true environmental state under a given set of radar configurations. It is this observed state that is used in the MDP and by the scan strategies in Section 4. If a storm cell p is scanned using a set of radar configurations S_r, the location, velocity, and radius attributes are observed as a function of the $U_p(p, S_r)$ quality defined in Section 3.1. The function $U_p(p, S_r)$ returns a value u between zero and one. Then the observed value of the attribute is the true value of the attribute plus some Gaussian noise distributed with mean zero and standard deviation $(1-u)V^{max}/\rho$ where V^{max} is the largest positive value the attribute can take and ρ is a scaling term. Since u depends on the decision epoch t, for the k-step look-ahead scan strategy we also use $\sigma_t = (1-u_t)V^{max}/\rho$ to compute the measurement error covariance matrix, R, in our Kalman filter.

We parameterize the MDP cost function as follows. Recall that the cost received for taking an action has three components: 1) the error between the true state and the observed state representing the quality of the observation, 2) whether all storm cells have been observed, and 3) whether all storm cells have been scanned at least once within the last T_r decision epochs. To compute the error attributed to 1) we sum over the difference in storm cell attribute values for the observed state and the true environment state. To compute the error attributed to 2) we assume that any unobserved storm cell has been observed with quality 0, hence $u = 0$. Summing over $(1-u)V^{max}/\rho$ for all attributes with $\sigma = 0$ gives the value $P_m = 15.5667$, and thus a penalty of 15.5667 is received for each unobserved storm cell. Finally, to compute the error attributed to 3) if a storm cell is not seen within $T_r = 4$ decision epochs a penalty of $P_r = 200$ is given. We use the value 200 to ensure that if a storm cell has not been rescanned within the appropriate amount of time, this part of the cost function will dominate.

Figure 3 summarizes how we simulate the environment, distinguishing the true state of the environment known only to the simulator from the observed state of the environment used in the MDP and by the other scan strategies. Note that the $k-$step look-ahead scan strategy maintains its own internal representation of the true state of the environment (through the use of Kalman filters), that is different from the true state of the environment in the simulator.

We distinguish between the true state of the environment and the observed state for several reasons. First, although radars provide measurements about meteorological phenomena, the true attributes of the phenomena are unknown. Poor overlap in a dual-Doppler area, scanning a subsector too quickly or slowly, or being unable to obtain a sufficient number of elevation scans will degrade the quality of the measurements. Consequently, models of previously existing phenomena may contain estimation errors such as incorrect velocity, propagating error into the future predicted locations of the phenomena. Second, when a radar scans a subsector, it obtains more accurate estimates of the phenomena in that subsector than if it had scanned a

Table 2: Parameter settings of program variables.

Variable	Meaning	Value
β	Weighting term in quality function	.5
κ_p	Decay rate of phenomenon quality	.25
κ_s	Decay rate of sector quality	.25
α	Learning rate for Sarsa(λ)	.0005
ϵ	Exploration rate for Sarsa(λ)	.01
γ	Discount factor for Sarsa(λ)	.9
λ	Eligibility decay for Sarsa(λ)	.3
ϕ	Discount factor for K-step strategy	.75
T_r	Number of decision epochs after which a storm must be scanned to avoid penalty	4
P_r	Penalty for not rescanning a storm within T_r decision epochs	200
P_m	Penalty for missing a storm	15.5667

full 360°, but less accurate estimates of the phenomena outside the subsector. Less accurate estimates may be due to a new phenomenon appearing in the unscanned area or due to an inaccurate predictive model for a previously observed phenomenon.

5.2 Results

In this section we present experimental results obtained using the simulation model of the previous section and the scan strategies described in Sections 3 and 4. Table 2 lists the parameter settings we used. Additionally, we use a single tiling for each of the x-location, y-location, x-velocity, y-velocity, radius, phenomenon confidence, and radar sector confidence state variables. For the (x, y) location and radius tilings, we use a granularity of 1.0; for the (x, y) velocity, phenomenon confidence, and radar sector confidence tilings, we use a granularity of 0.1. We first give an overview of the results and then discuss each set of plots in more detail.

Figures 4 to 8 examine the trade-offs among the sit-and-spin, myopic, k-step, and learned Sarsa(λ) scan strategies with respect to the following variables.

- *Measurement noise*: recall that the measurement noise was defined in Section 5.1 to be $\sim N(0, \sigma^2)$ where $\sigma = (1 - u)V^{max}/\rho$.
- *Radar radius*: we examine the effect of increasing the radar radius while keeping the maximum radius of a storm cell constant.
- *Maximum number of storm cells*: we examine the effect of increasing the maximum number of storm cells in the environment. As a consequence of the parameter settings of the simulation model, there may be storms that "arrive" but cannot be added to the state because the maximum number of storms would be exceeded. When there are a maximum of four storms, we restrict Sarsa(λ) to scanning only 180 or 360 degree sectors to reduce the time needed for convergence.
- *Prediction error*: defined to be the error between the predicted state for decision epoch t, S_t^{pred}, and the true state for decision epoch t, S_t^{true}. Note that for the sit-and-spin and myopic strategies, S_t^{pred} will equal their observed state at decision epoch $t - 1$.
- *Scan quality*: defined to be the quality U_p in Section 3.1. This measures the quality with which a phenomenon is scanned when it actually is observed (i.e., $F_c \geq 0.95$).
- *Cost*: for Figure 5(c), (d), defined to be the cost function C_1 in Equation (3). For Figure 6(d), defined to be the cost function C_2 in Equation (4). Recall that C_2 equals C_1 plus a penalty if a storm cell has not been scanned at least once within the last $T_r = 4$ epochs.

- *Re-scan interval*: defined to be the number of decision epochs to first scan or re-scan a storm cell.

We now discuss each set of plots in more detail, looking first at the differences in prediction error, then looking at the differences in scan quality, cost, and inter-scan time. Figure 4 shows the average difference in prediction error between the 2-step scan strategy and the sit-and-spin, myopic, and 1-step scan strategies for a 10 km (a) and 30 km (b) radar radius. We see that the 1-step and 2-step strategies have about the same prediction error, that the myopic strategy has higher error relative to the 2-step, and that the sit-and-spin strategy has the highest relative error. As measurement noise increases, the relative difference in prediction error between the 2-step versus the sit-and-spin and myopic strategies increases.

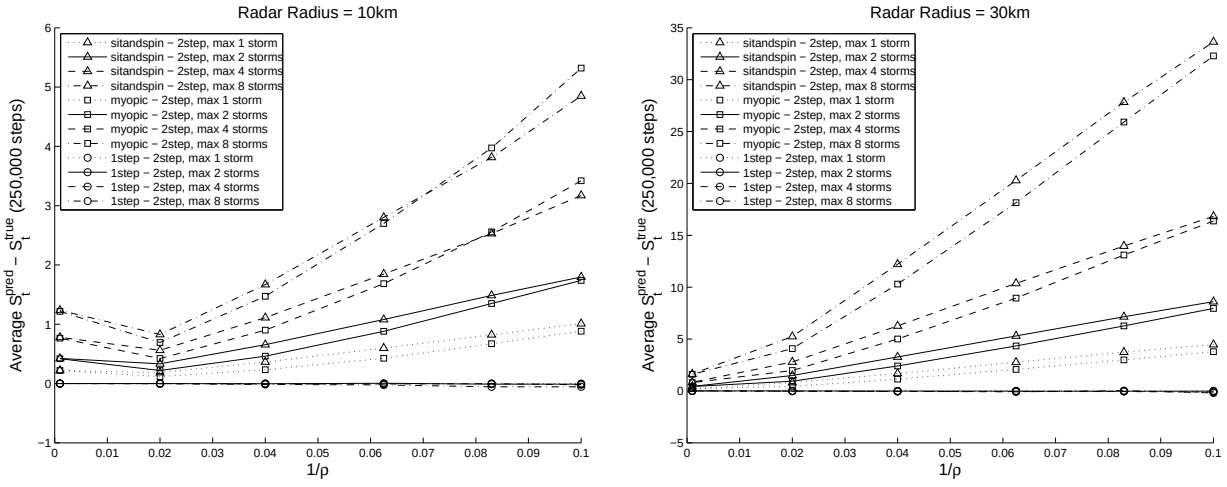
Figures 5(a) and (b) show the average difference in quality between the 2-step scan strategy and the sit-and-spin, myopic, and 1-step scan strategies for a 10 km (a) and 30 km (b) radar radius. For both the 10 km and 30 km radius, we see that the sit-and-spin strategy has the lowest scan quality relative to the 2-step, the myopic strategy has the next lowest relative quality, and the 1-step strategy has the highest relative quality. We also see that as the maximum number of storms in the environment increases from 1 to 8, the scan quality increases for the sit-and-spin: since there are now more possible storms, it is more likely that a storm cell is close to a radar. Thus, the F_d term of the quality function, see Section 3.1, is more likely to be large. Also notice that when there is at most one storm cell, the 1-step quality for scanning that storm cell is essentially the same as the 2-step quality. Finally, notice that decreasing the radar radius decreases the differences in quality of the different scan strategies, although the overall trends remain the same. We hypothesize that this is a consequence of the large maximum storm cell radius, 4 km, relative to the 10 km radar radius: larger scan sectors will be needed to fully cover any storm, thereby decreasing the scan quality. In summary, Figures 5(a) and (b) indicate that the 2-step strategy can slightly outperform the 1-step strategy in how well it scans storms, and more significantly outperforms the sit-and-spin and myopic strategies; and that the performance gain partly depends on the number of storms present in the environment and the size of maximum storm cell radius relative to the radar radius.

Figure 6(c) again examines scan quality, showing the average difference in quality between the learned Sarsa(λ) scan strategy and the sit-and-spin and 2-step strategies. We see that when $1/\rho = 0.001$ (i.e., little noise), Sarsa(λ) has the same or higher relative quality than does sit-and-spin, but significantly lower relative quality (0.05 to 0.15) than does the 2-step. This in part reflects the difficulty of learning to perform as well as or better than Kalman filtering. Examining the learned strategy we see that when there is at most one storm with observation noise $1/\rho = 0.001$, Sarsa(λ) learns to simply sit-and-spin, since sector scanning confers little benefit. As the observation noise increases, we see that the relative difference increases for the sit-and-spin strategy, and decreases for the 2-step strategy.

We now examine the differences in cost. Figures 5(c) and (d) show the average cost difference between the 2-step strategy and the sit-and-spin, myopic, and 1-step strategies for a 10 km (c) and 30 km (d) radar radius. The C_1 cost function is used here, see Equation (3); there is no penalty for not rescanning a storm within $T_r = 4$ decision epochs. For both the 10 km and 30 km radius, we see that the cost difference increases in favour of the 2-step strategy as the maximum number of storm cells increases and as the value of $1/\rho$ increases. When $1/\rho = 0.001$, there is little cost difference between the scan strategies. Finally, we see that as the radius is increased from 10 km to 30 km, the cost difference is larger, in part because the quality difference is larger.

Figure 6(d) again examines cost, showing the average difference in cost between the learned Sarsa(λ) scan strategy and the sit-and-spin and 2-step strategies for a 30 km radar radius. The C_2 cost function is used here, see Equation (4); there is now a penalty if a storm has not been scanned within the last $T_r = 4$ decision epochs. We see that when all scan strategies are penalized for not scanning a storm within $T_r = 4$ time-steps, Sarsa(λ) has the lowest average cost. For the C_1 cost function (not shown) the 2-step can perform as well as or better than Sarsa(λ). Together with Figure 5(d), this implies that the 2-step has more inter-scan times greater than $T_r = 4$ than does Sarsa(λ).

We now examine the differences in inter-scan time. Figure 7 shows the CDFs of the number of decision epochs before a storm cell is observed or re-scanned, for the sit-and-spin, 1-step, and 2-step strategies. We

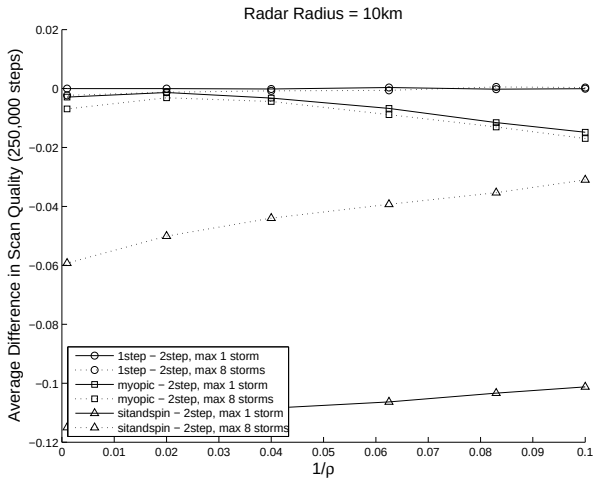


(a) (b)

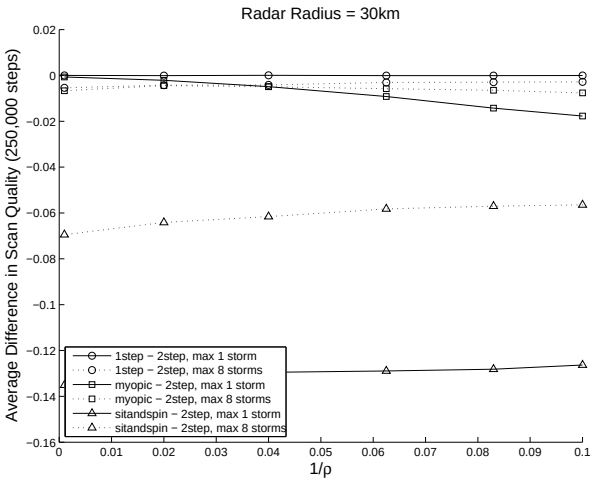
Figure 4: The dependence of state prediction error on measurement noise, radar radius, and maximum number of storms in the environment.

see from Figures 7(a) and (b), regardless of radar radius, that if there is at most 1 storm in the environment, and $1/\rho = 0.001$, the 1-step and k-step strategies rescan storms with approximately the same frequencies. We also see from Figures 7(a) and (b), that the inter-scan interval is typically lower for a 30 km radius than for a 10 km radius. We hypothesize that this is a consequence of the 4 km storm cell radius: since there is a maximum of 1 storm cell in the environment, a sector scan of a 30 km radius is more likely to cover at least 95% of the storm. From Figures 7(c) and (d) we see when there are at most 8 storm cells in the environment, the 1-step strategy rescans storm cells more quickly than does the 2-step, shown by the higher values taken on by the 1-step CDF. We also see from Figures 7(c) and (d), that the inter-scan interval is typically lower for a 10 km radius than for a 30 km radius, unlike the situation when there is at most 1 storm cell in the environment. We hypothesize that this is again a consequence of the 4 km storm cell radius: the size of the sector scans will be larger with a 10 km rather than a 30 km radius, consequently, multiple storms will more likely be covered, thereby decreasing the inter-scan time. Note that for the sit-and-spin CDF, $P[X \leq 1]$ is not 1; due to noise, for example, the measured location of a storm cell may be (expected) outside any radar footprint and consequently the storm cell will not be observed. Looking at the Sarsa(λ) inter-scan times, Figure 8 shows that, as a consequence of the penalty for not scanning a storm within $T_r = 4$ time-steps, while Sarsa(λ) may rescan fewer storm cells within 1, 2, or 3 decision epochs than do the other scan strategies, it scans almost all storm cells within 4 epochs.

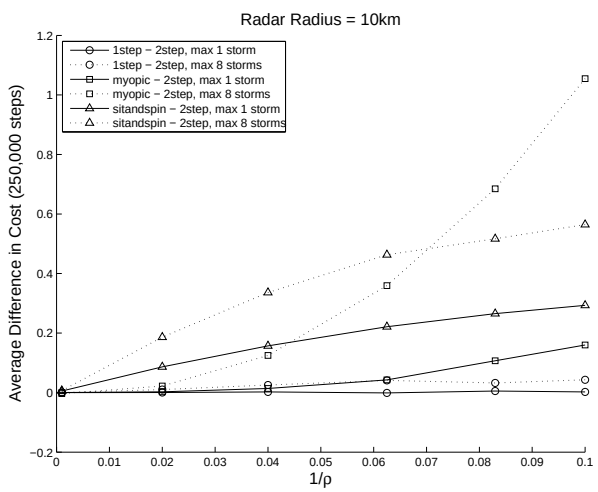
In conclusion, Figures 5 and 7 show that the 2-step strategy has higher scan quality and lower cost than the 1-step as the number of storms in the environment increases, but that the 1-step strategy rescans storms more frequently than does the 2-step. We hypothesize that as the maximum number of storms in the environment increases, the 2-step strategy focuses on individual storms more, thereby increasing the number of decision epochs between scans of the same storm. Figures 6 and 8 show that that there is a trade-off between inter-scan time and scan quality. We hypothesize that this trade-off occurs because increasing the size of the scan sectors ensures that inter-scan time is minimized, but decreases the scan quality.



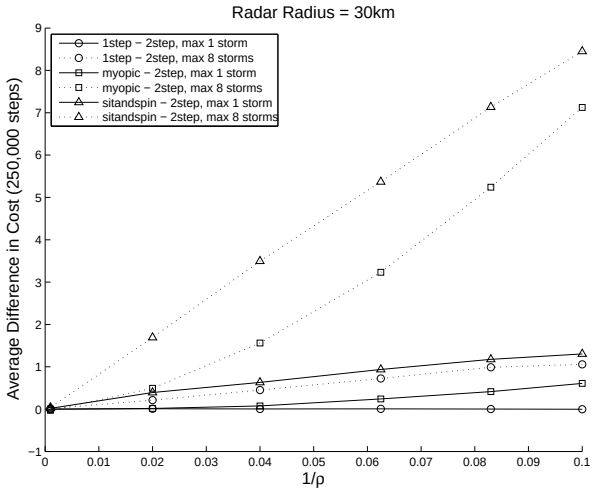
(a) Average difference in scan quality (relative to 2-step)



(b) Average difference in scan quality (relative to 2-step)



(c) Average difference in C_1 cost (relative to 2-step)

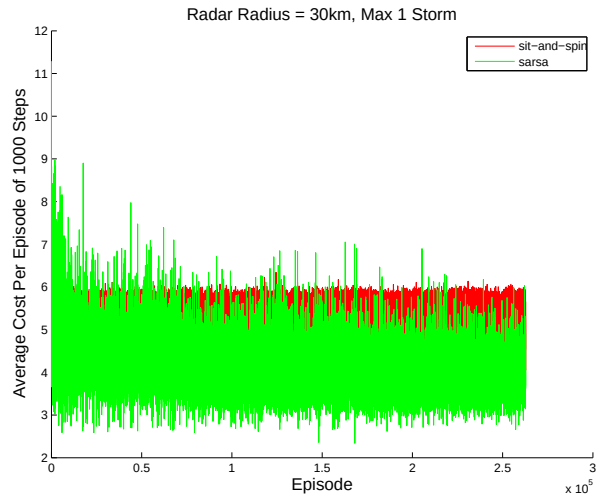


(d) Average difference in C_1 cost (relative to 2-step)

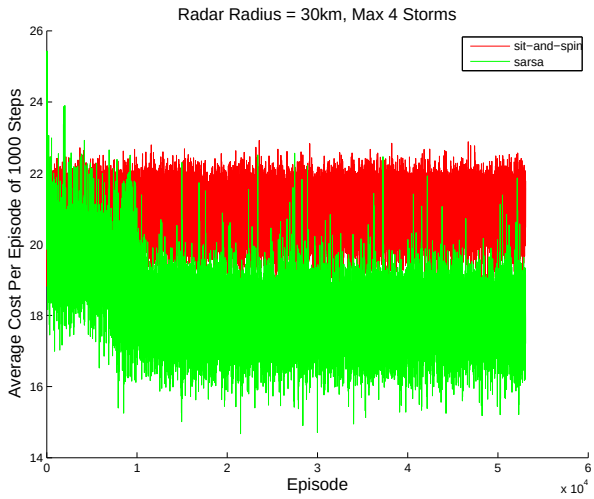
Figure 5: The dependence of scan quality (a), (b), and cost (c), (d), on measurement noise, radar radius, and maximum number of storms in the environment.

6 Related Work

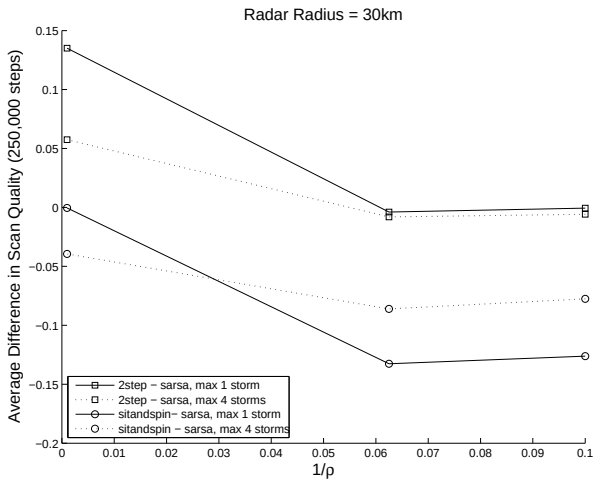
Work by [6] examines the problem of lookahead scheduling of agile radars on airplanes for detecting and tracking ground targets. They show that lookahead scan strategies for radar tracking of a ground target outperform myopic strategies. In comparison, we consider the problem of tracking meteorological phenomena using ground radars. [6] uses an information theoretic measure to define the reward metric and then proposes both an approximate solution to solving the MDP Bellman equations as well as a reinforcement learning-based solution to obtain a lookahead policy. We note that [6] uses an off-policy reinforcement learning algorithm Q-learning, while we use an on-policy algorithm Sarsa(λ). Off-policy algorithms update the action-value function using the currently maximal action, while on-policy algorithms use the action that was actually



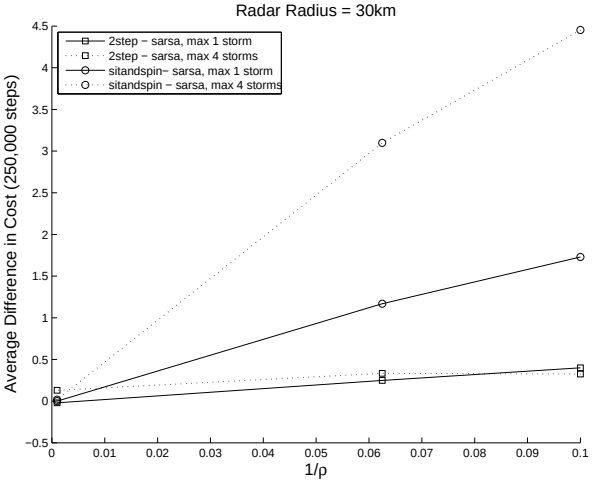
(a)



(b)



(c) Average difference in scan quality (relative to Sarsa)



(d) Average difference in C_2 cost (relative to Sarsa)

Figure 6: (a) and (b) show example convergences of Sarsa(λ). (c) and (b) show the dependence of scan quality and cost on measurement noise, radar radius, and maximum number of storms in the environment.

executed; this has implications for function approximation, please see [18, 12] for further information.

Work by [20] examines where to target the radar beams and which waveform to use for electronically steered phased array radars. They maintain a set of error covariance matrices and dynamical models for existing targets, as well as track existence probability density functions to model the probability that targets appear. They then choose the scan mode for each target that has both the longest revisit time for scanning a target and error covariance below a threshold. They do this for scheduling 1-step and 2-steps ahead and show that considering the environment two decision epochs ahead outperforms a one-step look-ahead for tracking of multiple targets.

Within sensor networks, [8] examine the use of game theory and reinforcement learning to allocate resources

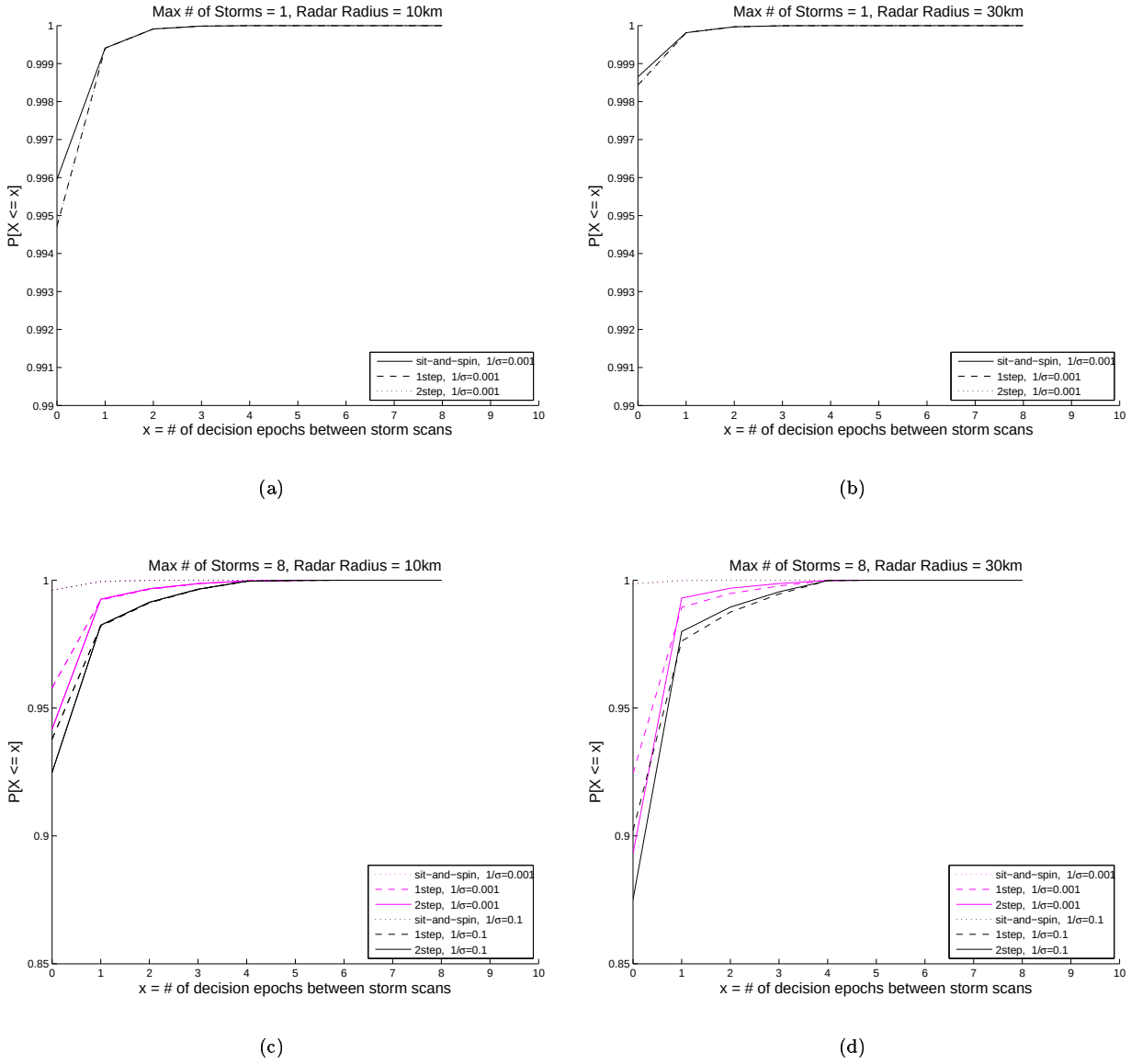
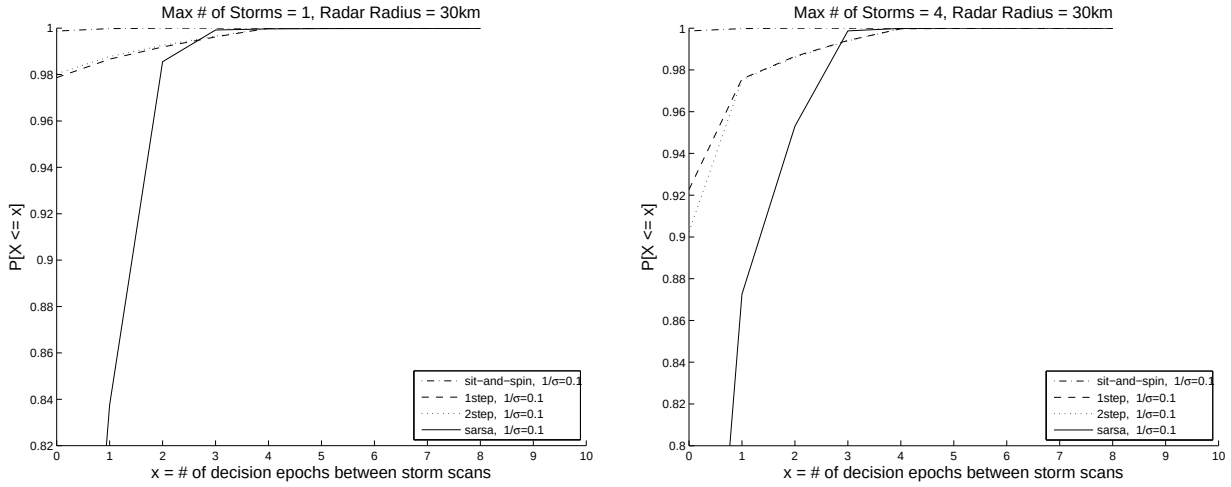


Figure 7: The dependence of inter-scan time on measurement noise, radar radius, and maximum number of storms in the environment.

in a sensor network. They focus on actions, using reinforcement learning to learn the profit associated with different actions, rather than the profit associated with different state-action pairs. Besides sensor networks, other reinforcement learning applications in large state spaces include robot soccer [15], helicopter control [10] and planetary rovers [23].

7 Conclusions and Future Work

In this work we have compared the performance of myopic and lookahead scan strategies in the context of the meteorological radar scheduling problem. We showed that the main benefits of using a lookahead



(a) (b)

Figure 8: The dependence of inter-scan time on measurement noise, radar radius, and maximum number of storms in the environment.

strategy are when there are multiple meteorological phenomena in the environment, and when the maximum radius of any phenomenon is sufficiently smaller than the radius of the radars. We also showed that there is a trade-off between the average quality with which a phenomenon is scanned and the number of decision epochs before which a phenomenon is rescanned.

There are two main directions for future work. The first involves addressing the time complexity issues associated with the continuous state and action spaces. The second involves incorporating more radar and meteorological information into the transition, measurement, and cost functions; ideally, we would like to use real meteorological data rather than a simulated environment.

Rather than identifying a policy that chooses the best action to execute in a state for a single decision epoch, as in this work, it may be useful to consider actions that cover multiple decision epochs, as in semi-Markov decision processes [19]. Similarly, controllers as used in robotics [4], could be useful. This is one method by which prior information, for instance, scanning preferences of different meteorologists, could be incorporated into the scanning strategy, and would also reduce the convergence time required by the reinforcement learning algorithm to identify a good policy.

8 Acknowledgments

This work was supported in part by the National Science Foundation under under the Engineering Research Centers Program, Award number EEC-0313747. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the National Science Foundation.

References

[1] D. Cox and V. Isham. A simple spatial-temporal model of rainfall. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 415:1849:317–328, 1988.

[2] B. Donovan and D. J. McLaughlin. Improved radar sensitivity through limited sector scanning: The dcas approach. In *Proceedings of AMS Radar Meteorology*, 2005.

[3] K. Hondl. Capabilities and components of the warning decision and support system - integrated information (WDSS-II). In *Proc. American Meteorological Society Annual Meeting*, January 2003.

[4] M. Huber and R. Grupen. A feedback control structure for on-line learning tasks. *Robotics and Autonomous Systems*, 22(3-4):303–315, 1997.

[5] W. T. M. III, F. Glanz, and L. G. K. III. CMAC: An associative neural network alternative to backpropagation. *Proceedings of the IEEE*, 78:10:1561–1567, 1990.

[6] C. Kreucher and A. O. H. III. Non-myopic approaches to scheduling agile sensors for multistage detection, tracking and identification. In *Proceedings of ICASSP*, pages 885–888, 2005.

[7] C. Kwok and D. Fox. Reinforcement learning for sensing strategies. In *IROS*, 2004.

[8] G. Mainland, D. Parkes, and M. Welsh. Decentralized, adaptive resource allocation for sensor networks. In *NSDI*, 2005.

[9] K. Murphy. A survey of POMDP solution techniques. Technical Report Technical Report, U.C. Berkeley, 2000.

[10] A. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, and E. Liang. Inverted autonomous helicopter flight via reinforcement learning. In *International Symposium on Experimental Robotics*, 2004.

[11] D. Pepyne et al. Defining and optimizing utility in NetRad, a collaborative adaptive sensor network for hazardous weather detection, CASA technical report. 2006.

[12] T. Perkins and D. Precup. A convergent form of approximate policy iteration. In *NIPS*, 2002.

[13] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *In Proceedings of IJCAI*, 2003.

[14] I. Rodrigues-Iturbe and P. Eagleson. Mathematical models of rainstorm events in space and time. *Water Resources Research*, 23:1:181–190, 1987.

[15] P. Stone, R. Sutton, and G. Kuhlmann. Reinforcement learning for robocup-soccer keepaway. *Adaptive Behavior*, 3, 2005.

[16] R. Sutton. Tile coding software. *http://rlai.cs.ualberta.ca/RLAI/RLtoolkit/tiles.html*.

[17] R. Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *NIPS*, 1996.

[18] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, Massachusetts, 1998.

[19] R. Sutton, D. Precup, and S. Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112:181–211, 1999.

[20] S. Suvorova, D. Musicki, B. Moran, S. Howard, and B. L. Scala. Multi step ahead beam and waveform scheduling for tracking of manoeuvering targets in clutter. In *Proceedings of ICASSP*, pages 889–892, 2005.

[21] J. M. Trabal, B. C. Donovan, M. Vega, V. Marrero, D. J. McLaughlin, and J. G. Colom. Puerto Rico student test bed applications and system requirements document development. In *Proceedings of the 9th International Conference on Engineering Education*, 2006.

[22] G. Welch and G. Bishop. An introduction to the Kalman filter. Technical Report TR95-041, U of North Carolina at Chapel Hill, Department of Computer Science, 1995.

[23] S. Zilberstein, R. Washington, D. Bernstein, and A. Mouaddib. Decision-theoretic control of planetary rovers. In *Plan-Based Control of Robotic Agents, LNAI*, 2002.

[24] M. Zink, D. Westbrook, S. Abdallah, B. Horling, V. Lakamraju, E. Lyons, V. Manfredi, J. Kurose, and K. Hondl. Meteorological command and control: An end-to-end architecture for a hazardous weather detection sensor network. *Workshop on End-to-End, Sense-and- Respond Systems, Applications, and Services*, 2005.