

SMART SLEEPING POLICIES FOR WIRELESS SENSOR NETWORKS

Venu Veeravalli

ECE Dept and Coordinated Science Lab
University of Illinois at Urbana-Champaign

`vvv@uiuc.edu`

`http://www.comm.csl.uiuc.edu/~vvv`

(Joint work with Jason Fuemmeler, Arun Visvanathan)

NSF Workshop on Future Directions in Systems Research
for Networked Sensing, May 25, 2006

Ways to Save Energy in Wireless Sensor Networks

- Efficient source coding
- Efficient Tx/Rx design
- Efficient processor design
- Power control
- Efficient routing
- **Switching nodes between active and sleep modes**

Active \iff Sleep Transition

- **External Activation**
 - Low-power **paging** channel to **wake up** sensors when needed
 - **But** power consumed by paging channel is usually not negligible compared to power consumed by sensor in active state
 - Passive RF-ID technology?
- **Practical Assumption:** Sensor that is asleep **cannot** be communicated with or woken up prematurely
 - \implies sleep duration has to be chosen when sensor goes to into sleep mode
- Having sleeping sensors could result in communication/sensing **performance degradation**
- **Design Problem:** Find sleeping policies that optimize **tradeoff** between energy consumption and performance

Sleeping Policies

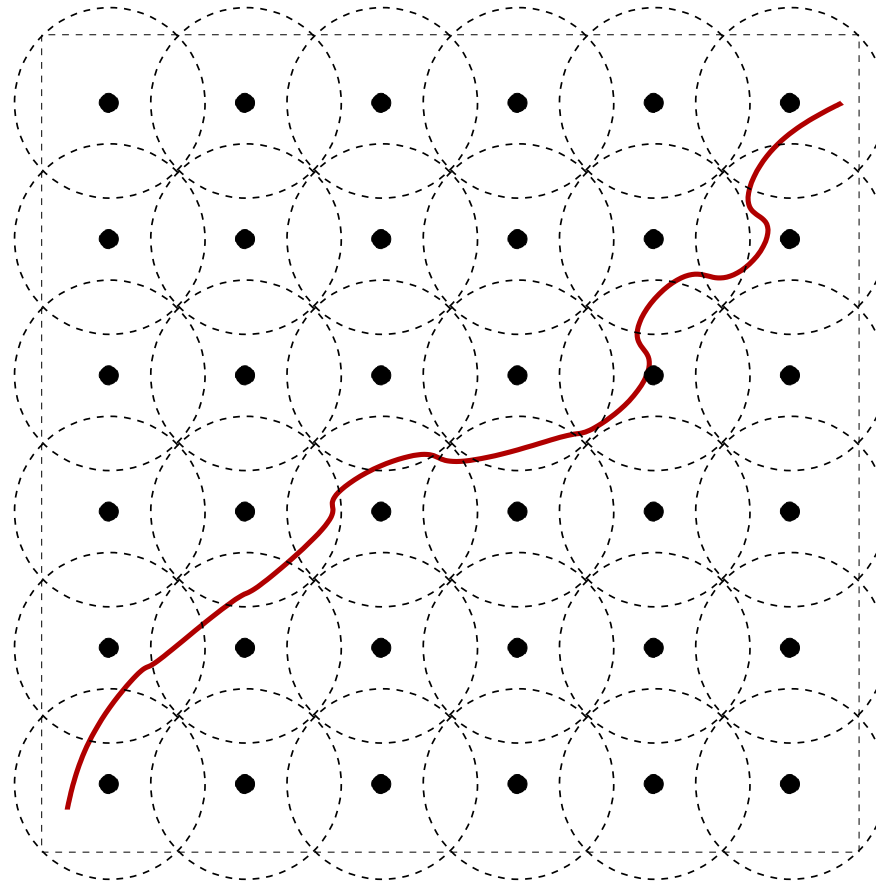
- **Duty Cycle Policies**

- Sensor sleeps with deterministic or random (with predetermined statistics) duty cycle
- Synchronous or asynchronous across sensors
- Duty cycle chosen to provide desired tradeoff between energy and performance
- Simple to implement, generic

- **Smart (Adaptive) Policies**

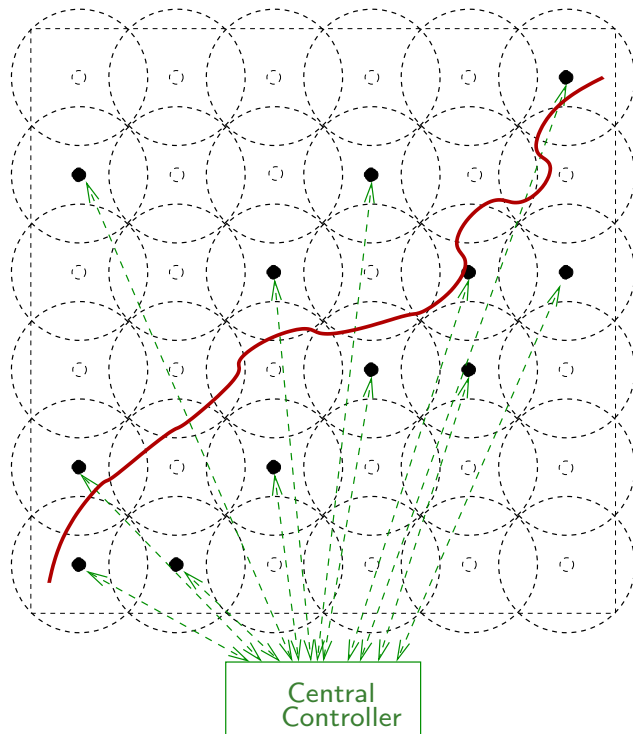
- Use all available information about the state of the sensor system to set sleep time of sensor
- Application specific \implies **system-theoretic** approach required
- **Potential energy savings over duty cycle policies**

Tracking Using Dense Network of Wireless Sensors



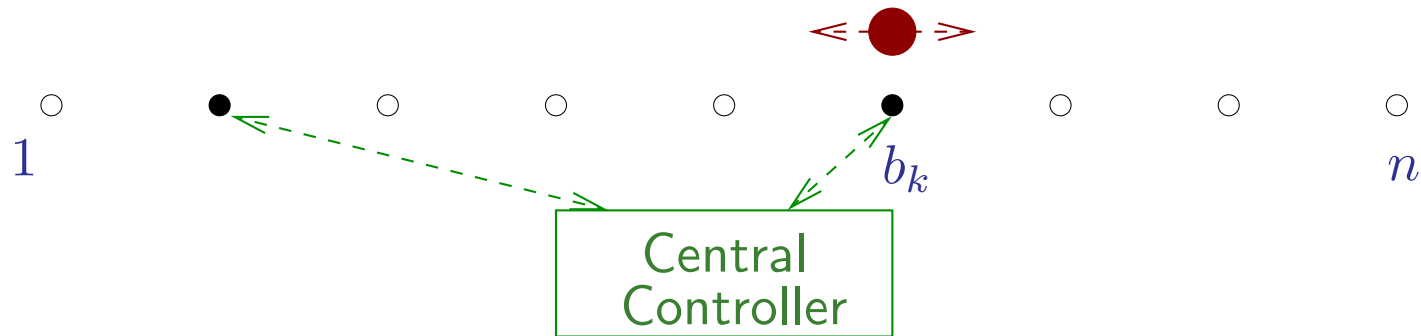
- Sensor detects presence of object within close vicinity
- Sensors switch between **active** and **sleep** modes to save energy – sensors need to come awake when needed for tracking

General Problem Description



- Sensors distributed in **two-dimensional** field
 - Sensor that is awake detects object **without error** within range
 - Sensor field is sufficiently **dense** so that sensors cover area when all awake
 - Object follows **random** path whose statistics are assumed to be known
 - **Central controller** communicates with sensors that are awake
-
- Sensor that wakes up remains awake for **one** time unit, during which it:
 - reports detecting object if object is in its range
 - receives new sleep time from central controller
 - sets its sleep timer to new sleep time

One-Dimensional Tracking Problem



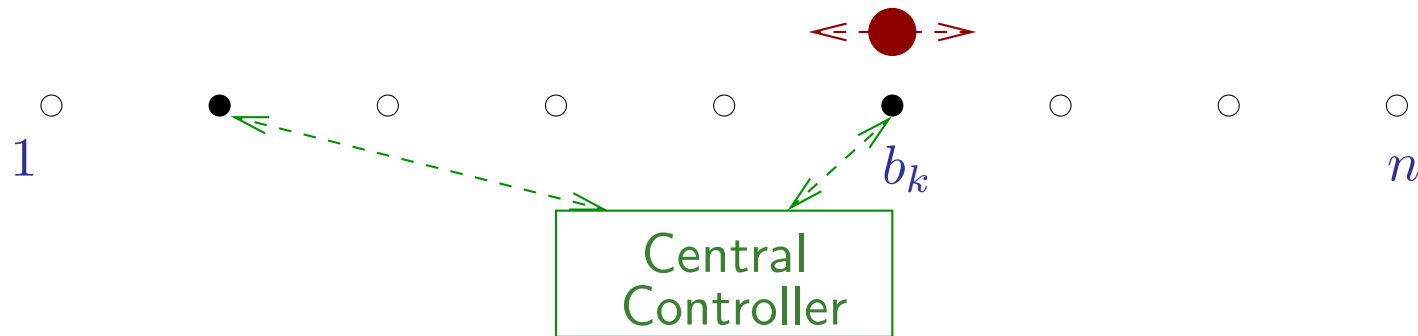
- Discrete-time, discrete-space
- b_k = position of object at time k

$$b_{k+1} = b_k + w_k$$

- $r_{k,\ell}$ = residual sleep time at sensor ℓ at time k
- $u_{k,\ell}$ = control input (sleep time) at sensor ℓ at time k

$$r_{k+1,\ell} = (r_{k,\ell} - 1)\mathbb{1}_{\{r_{k,\ell} > 0\}} + u_{k,\ell}\mathbb{1}_{\{r_{k,\ell} = 0\}}$$

Markov Decision Process (MDP)

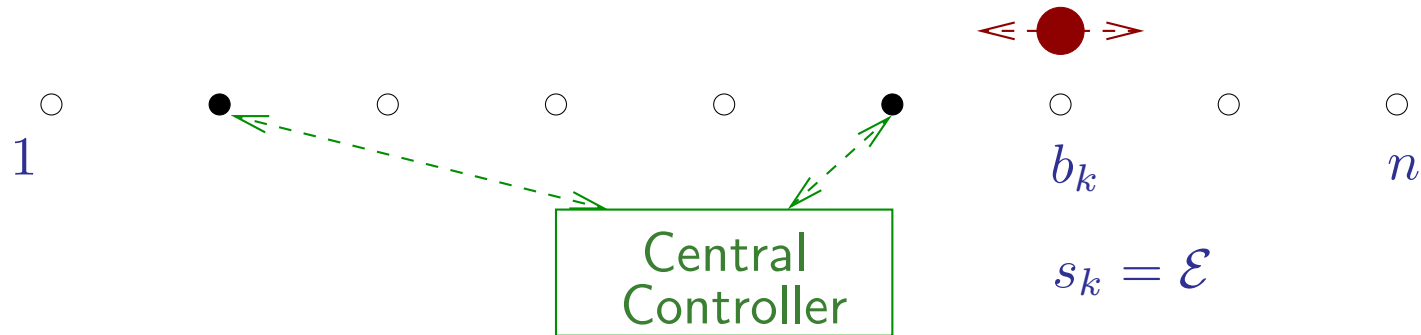


- Dynamical system with **state** $x_k = (b_k, \mathbf{r}_k)$ and possible **termination**

$$x_{k+1} = \begin{cases} f(x_k, \mathbf{u}_k, w_k) & \text{if } x_k \neq \mathcal{T} \\ \mathcal{T} & \text{if } x_k = \mathcal{T} \text{ or if } b_k \notin \{1, \dots, n\} \end{cases}$$

- Exogenous input w_k and control input \mathbf{u}_k

Partially Observable State



- Yields partially observable MDP (POMDP)
- Observation $z_k = (s_k, \mathbf{r}_k)$

$$s_k = \begin{cases} b_k & \text{if } r_{k,b_k} = 0 \\ \mathcal{E} & \text{if } r_{k,b_k} > 0 \end{cases}$$

- Information available for decision-making at time k

$$I_k = (z_0, \dots, z_k, \mathbf{u}_0, \dots, \mathbf{u}_{k-1})$$

Cost Function and Optimization Problem

- Control function: If $r_{k,\ell} = 0$, $u_{k,\ell} = \mu_{k,\ell}(I_k)$
- Cost at time k : For $x_k \neq \mathcal{T}$,

$$g(x_k) = c \mathbb{1}_{\{r_{k,b_k} > 0\}} + \sum_{\ell=1}^n \mathbb{1}_{\{r_{k,\ell} = 0\}}$$

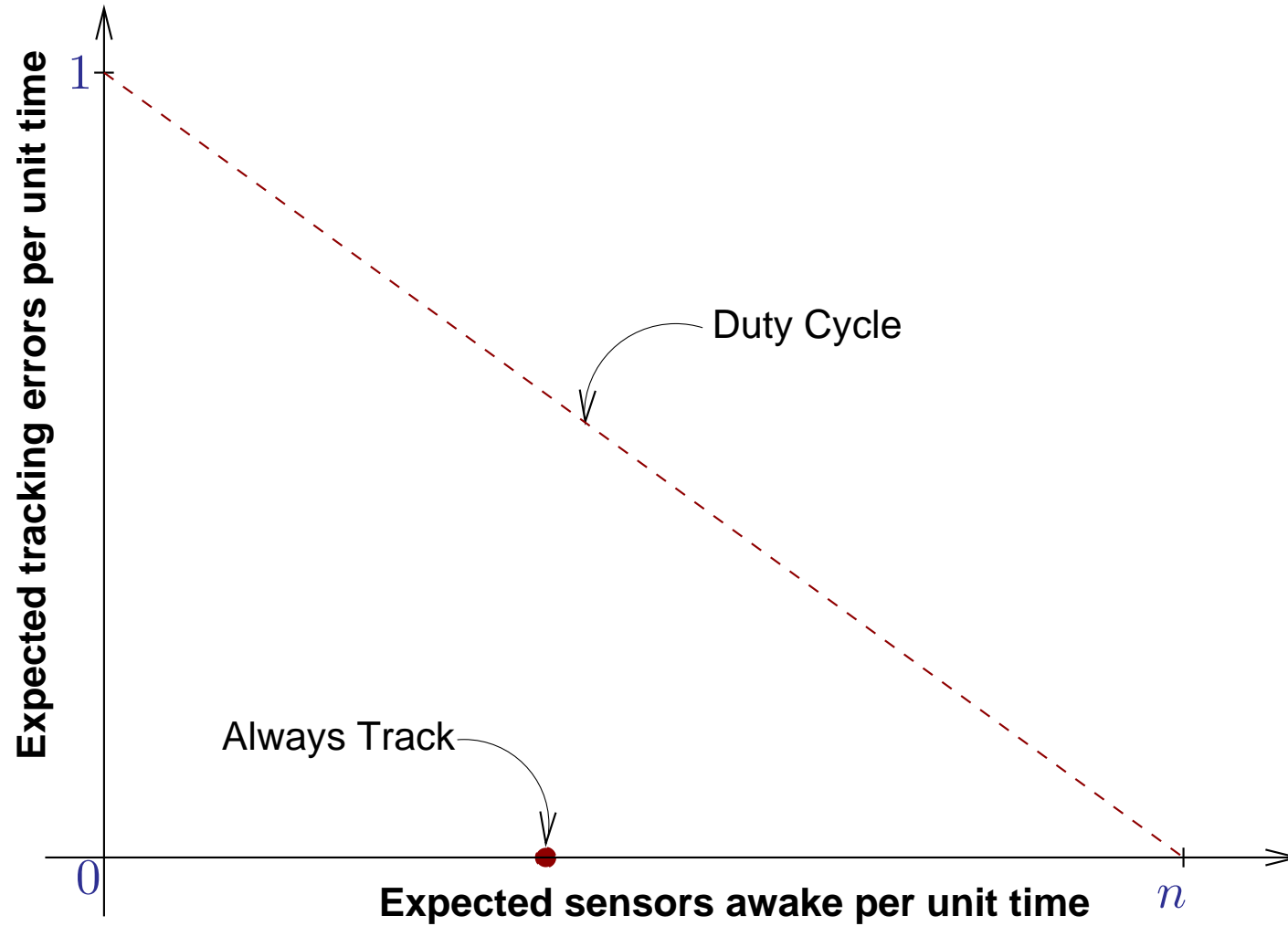
- Total expected cost

$$J(I_0, \mu_0, \mu_1, \dots) = \mathbb{E} \left[\sum_{k=1}^{\infty} g(x_k) \middle| I_0 \right]$$

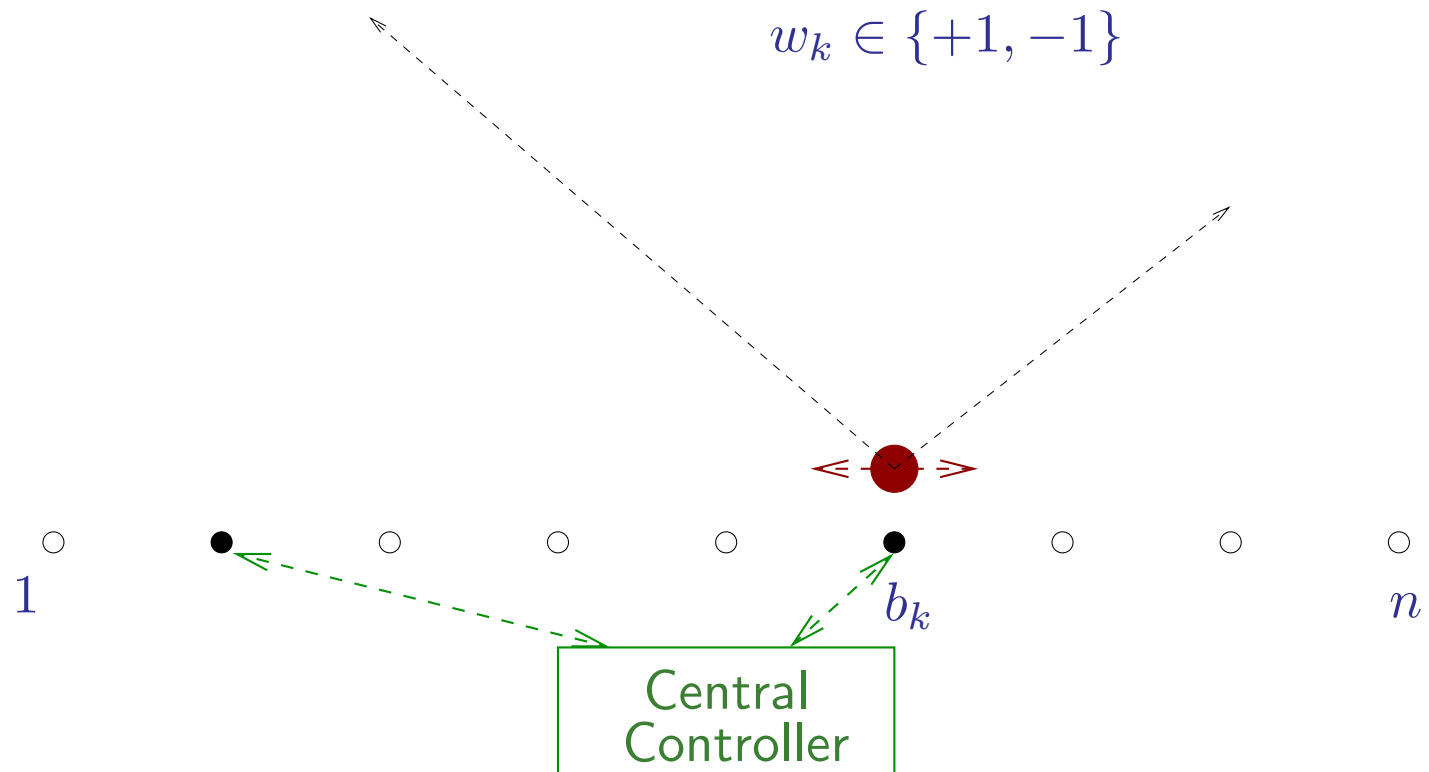
- Optimization Problem

$$J^*(I_0) = \min_{\mu_0, \mu_1, \dots} J(I_0, \mu_0, \mu_1, \dots)$$

Duty Cycle and Always Track Policies



Always Track Policy



Cost Function and Optimization Problem

- Control function: If $r_{k,\ell} = 0$, $u_{k,\ell} = \mu_{k,\ell}(I_k)$
- Cost at time k : For $x_k \neq \mathcal{T}$,

$$g(x_k) = c \mathbb{1}_{\{r_{k,b_k} > 0\}} + \sum_{\ell=1}^n \mathbb{1}_{\{r_{k,\ell} = 0\}}$$

- Total expected cost

$$J(I_0, \mu_0, \mu_1, \dots) = \mathbb{E} \left[\sum_{k=1}^{\infty} g(x_k) \middle| I_0 \right]$$

- Optimization Problem

$$J^*(I_0) = \min_{\mu_0, \mu_1, \dots} J(I_0, \mu_0, \mu_1, \dots)$$

Sufficient Statistic for Dynamic Programming

- Information I_k becomes unbounded in memory as k increases
 - Probability distribution of $x_k = (b_k, \mathbf{r}_k)$ given I_k is a sufficient statistic (a.k.a. belief state)
 - Part of state (i.e., \mathbf{r}_k) is observable
- ⇒ **Sufficient statistic** is given by $v_k = (\mathbf{p}_k, \mathbf{r}_k)$, where \mathbf{p}_k is the probability distribution of the object location given I_k

$$p_{k,\ell} = \mathbb{P}(\{b_k = \ell\} | I_k), \quad \ell = 1, \dots, n,$$

$$p_{k,n+1} = \mathbb{P}(\{b_k = \mathcal{T}\} | I_k)$$

Recursive Update for \mathbf{p}_k

- **Step 1:** Update \mathbf{p}_k without using new observation z_{k+1} , i.e., only using I_k

$$q_{k+1,\ell} = \mathbb{P}(\{b_{k+1} = \ell\} | I_k)$$

Markov evolution with transition matrix \mathbb{P} obtained from random walk statistics

$$\mathbf{q}_{k+1} = \mathbf{p}_k \mathbb{P}$$

- **Step 2:** "Clean up" \mathbf{q}_{k+1} to get \mathbf{p}_{k+1} using new observation z_{k+1}
 - Object position known $\implies \mathbf{p}_{k+1}$ point mass distribution
 - Otherwise, zero out \mathbf{q}_{k+1} where we know the object is not and renormalize

Optimal Solution via Dynamic Programming

- Can write down dynamic programming equations to solve optimization problem and find **Bellman equation**
 - However, even with sufficient statistic $v_k = (\mathbf{p}_k, \mathbf{r}_k)$, state space grows **exponentially** with number of sensors
- ⇒ DP solution is **not tractable** even for relatively small networks

Cost Function and Optimization Problem

- Control function: If $r_{k,\ell} = 0$, $u_{k,\ell} = \mu_{k,\ell}(I_k)$
- Cost at time k : For $x_k \neq \mathcal{T}$,

$$g(x_k) = c \mathbb{1}_{\{r_{k,b_k} > 0\}} + \sum_{\ell=1}^n \mathbb{1}_{\{r_{k,\ell} = 0\}}$$

- Total expected cost

$$J(I_0, \mu_0, \mu_1, \dots) = \mathbb{E} \left[\sum_{k=1}^{\infty} g(x_k) \middle| I_0 \right]$$

- Optimization Problem

$$J^*(I_0) = \min_{\mu_0, \mu_1, \dots} J(I_0, \mu_0, \mu_1, \dots)$$

Genie-Aided MDP Solution

- Genie gives us b_k
- Additive cost structure across sensors
 - \implies optimization problem decouples into n separate problems
 - \implies r_k not necessary for picking optimal $\mu_{k,\ell}$
 - \implies state is simply b_k (state space grows linearly with # sensors)

- Bellman equation

$$\tilde{J}_\ell(b_0) = \min_{\mu_\ell} \mathbf{E} \left[c \sum_{i=1}^u \mathbb{1}_{\{b_i=\ell\}} + \mathbb{1}_{\{b_{u+1} \neq \mathcal{T}\}} + \tilde{J}_\ell(b_{u+1}) \right] \Big|_{u=\mu_\ell(b_0)}$$

- Solved easily using policy iteration to yield stationary optimal policy μ_ℓ^*

Suboptimal Solutions for POMDP Problem

- Optimization problem **decouples** into n **separate** problems under certain simplifying assumptions
 - Assume no future observations \implies **FCR solution**
 - Assume perfect future observations \implies **Q_{MDP} solution**
- One problem for each sensor
- Only times of interest are those when sensor comes awake \implies r_k not needed
- **Bonus:** Assuming perfect future observations yields a **lower bound** on **optimal** performance

First Cost Reduction (FCR) Solution

- Assume no future observations
- Bellman equation:

$$J_\ell(\mathbf{p}) = \min_{\mu_\ell} \left(\sum_{i=1}^u c [\mathbf{p}^{\mathbb{P}^i}]_\ell + \sum_{j \neq \mathcal{I}} [\mathbf{p}^{\mathbb{P}^{u+1}}]_j + J_\ell(\mathbf{p}^{\mathbb{P}^{u+1}}) \right) \Big|_{u=\mu_\ell(\mathbf{p})}$$

- FCR solution: Choose first value of u such that

$$c [\mathbf{p}^{\mathbb{P}^{u+1}}]_\ell \geq \sum_{j \neq \mathcal{I}} [\mathbf{p}^{\mathbb{P}^{u+1}}]_j$$

Q_{MDP} Solution

- Name comes from POMDP literature
- Assume perfect future observations
- Bellman equation:

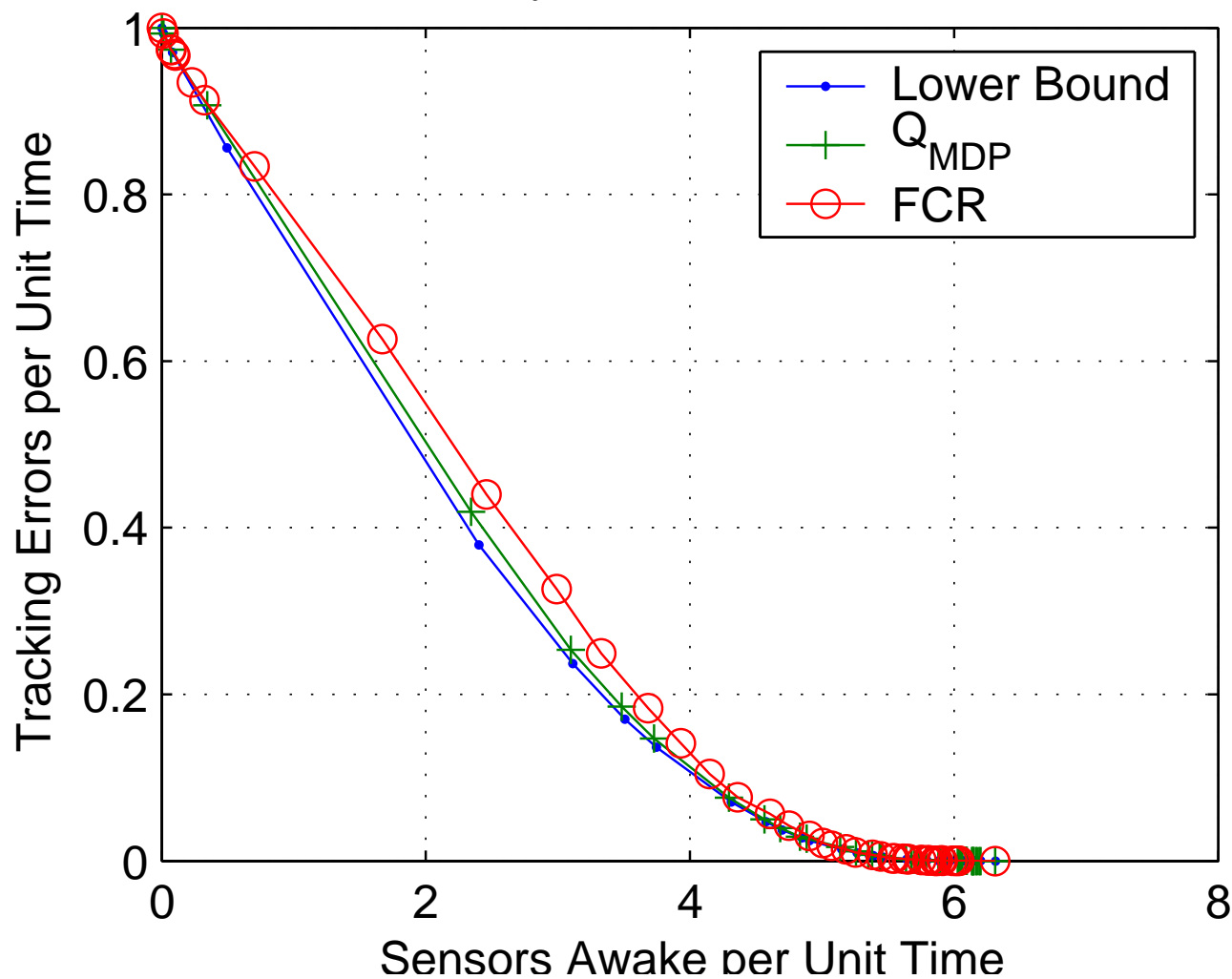
$$J_\ell(\mathbf{p}) = \min_{\mu_\ell} \left(\sum_{i=1}^u c [\mathbf{pP}^i]_\ell + \sum_{j \neq \mathcal{T}} [\mathbf{pP}^{u+1}]_j + \sum_j [\mathbf{pP}^{u+1}]_j J_\ell(\mathbf{e}_j) \right) \Big|_{u=\mu_\ell(\mathbf{p})}$$

- Q_{MDP} solution: Solve Bellman equation for genie-aided MDP to define the policy, $J_\ell(\mathbf{e}_j) = \tilde{J}_\ell(j)$
- Cost obtained is a lower bound on optimal performance

Results for Symmetric Random Walk

41 sensors, w_k i.i.d. Bernoulli(0.5) on $\{-1, +1\}$

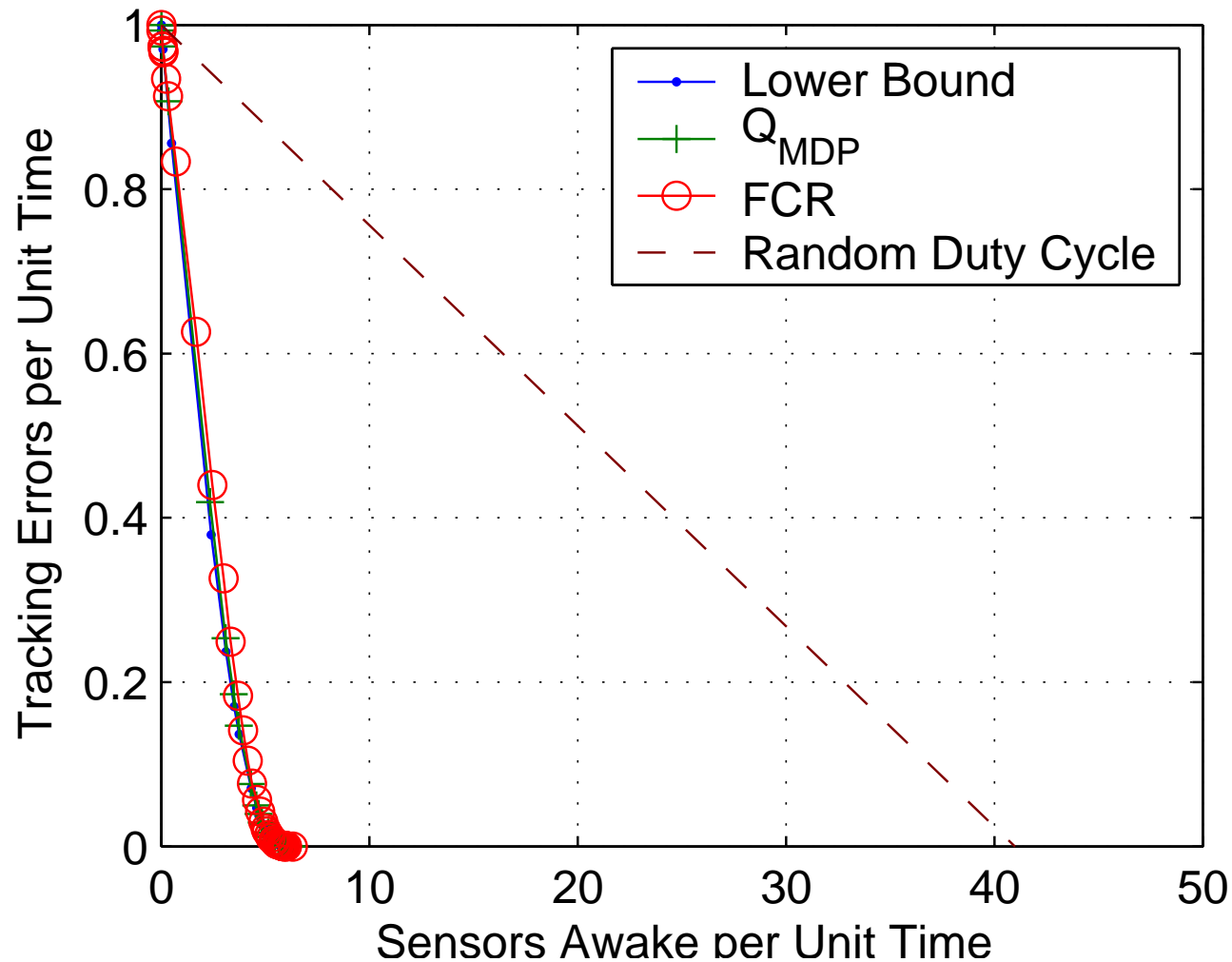
$m=20$, Symmetric Random Walk



Comparison with Random Duty Cycle Scheme

41 sensors, w_k i.i.d. Bernoulli(0.5) on $\{-1, +1\}$

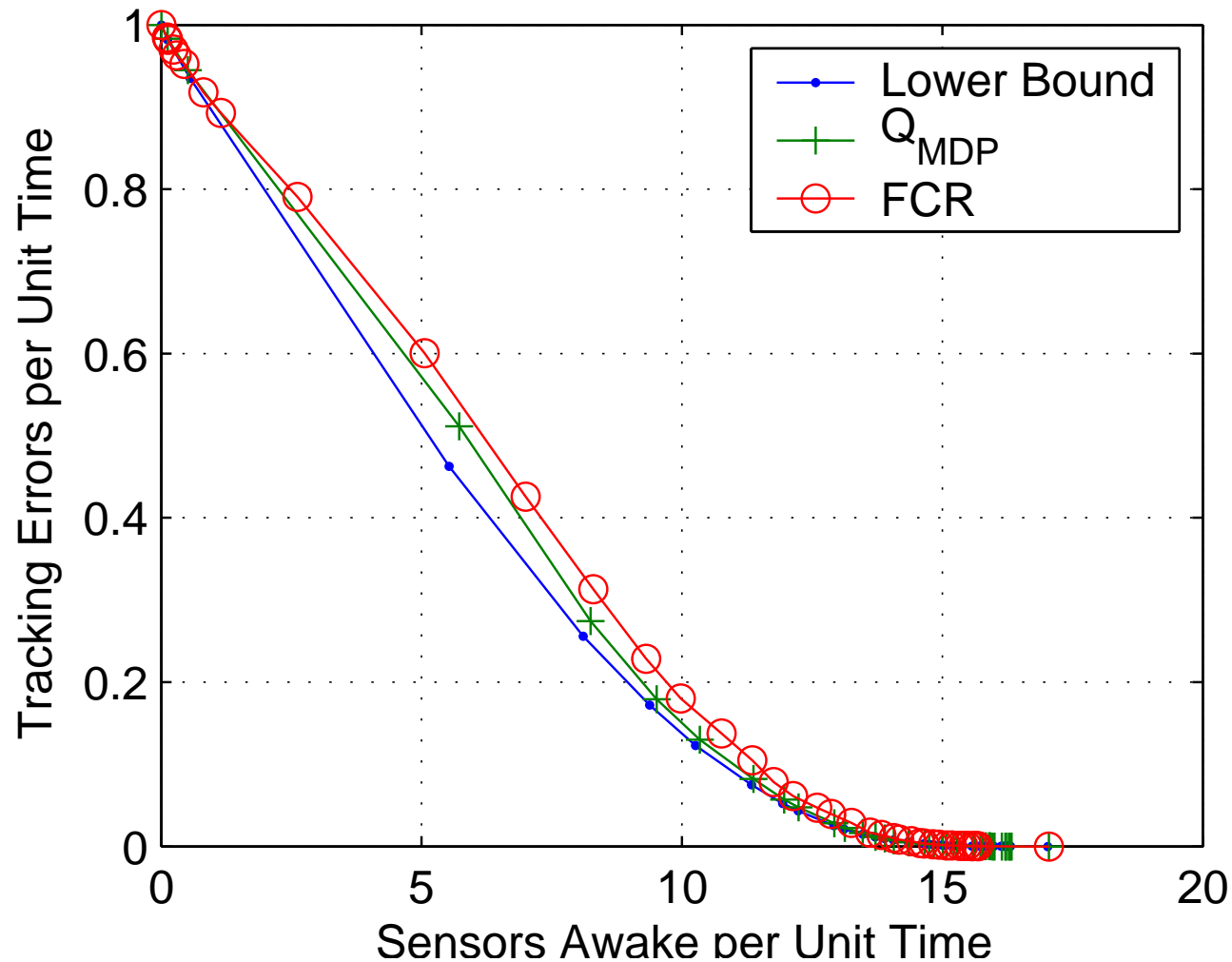
$m=20$, Symmetric Random Walk



Results for Three-Step Random Walk

61 sensors, w_k i.i.d. Uniform on $\{-3, -2, -1, 0, 1, 2, 3\}$

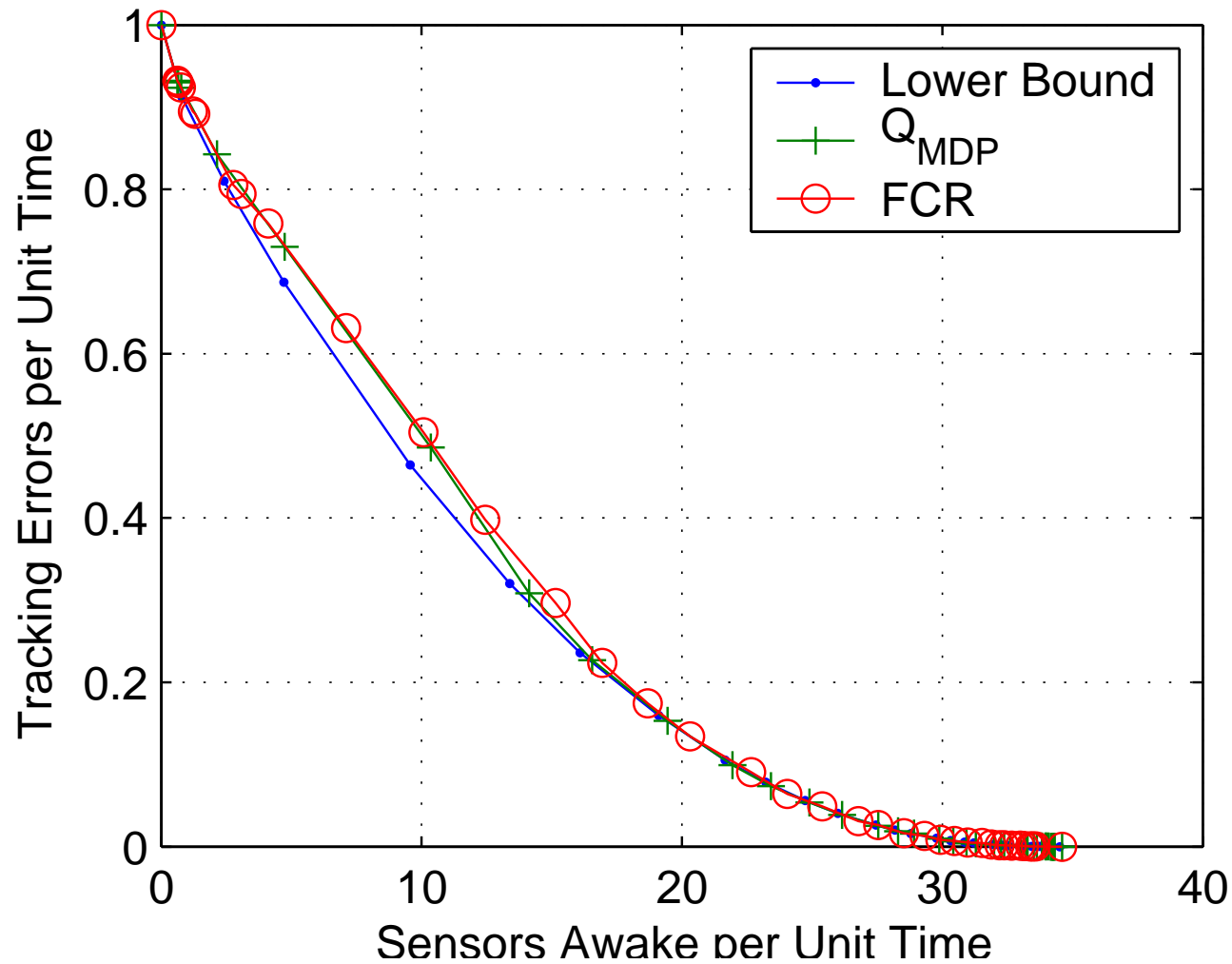
$m=30, \pm 3$ Max Uniform



Results for Two-Dimensional Random Walk

121 sensors, w_k i.i.d. Uniform on 3×3 Grid

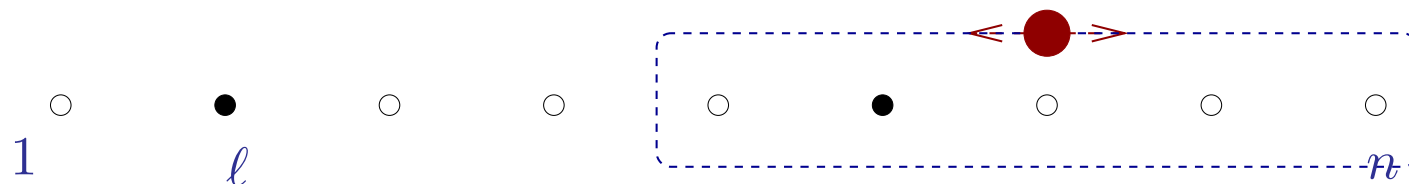
11×11 Square, Uniform on 3×3 Grid



Off-line Computation of Suboptimal Policies

- Can compute policies on-line, but this requires sufficient processing power and could introduce delays
- Policies need to be computed for each sensor location and each possible value for p
 - ⇒ storage requirements for off-line computation may be immense for large networks
- Off-line computation is feasible if we replace p with a point mass distribution
- Storage required is n values per sensor

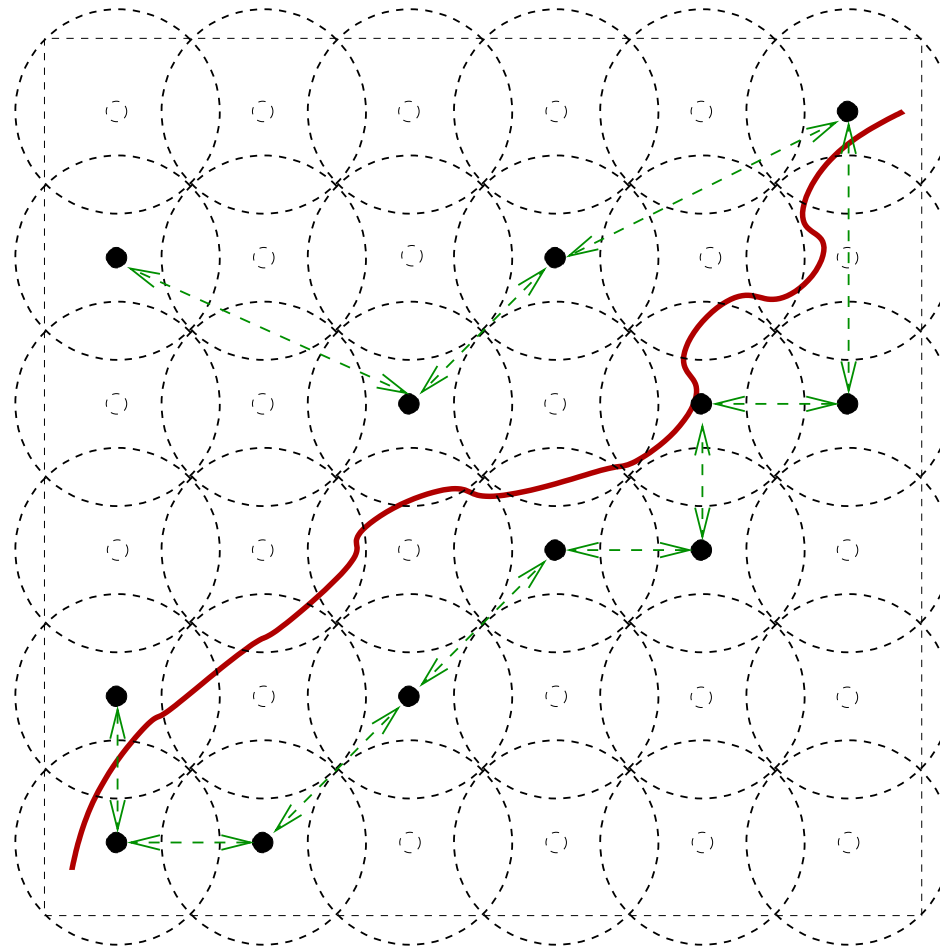
Point Mass Approximation for p



- Two options for placing point mass:
 - Centroid of distribution p
 - Nearest point to sensor on support of p

Decentralized Implementation

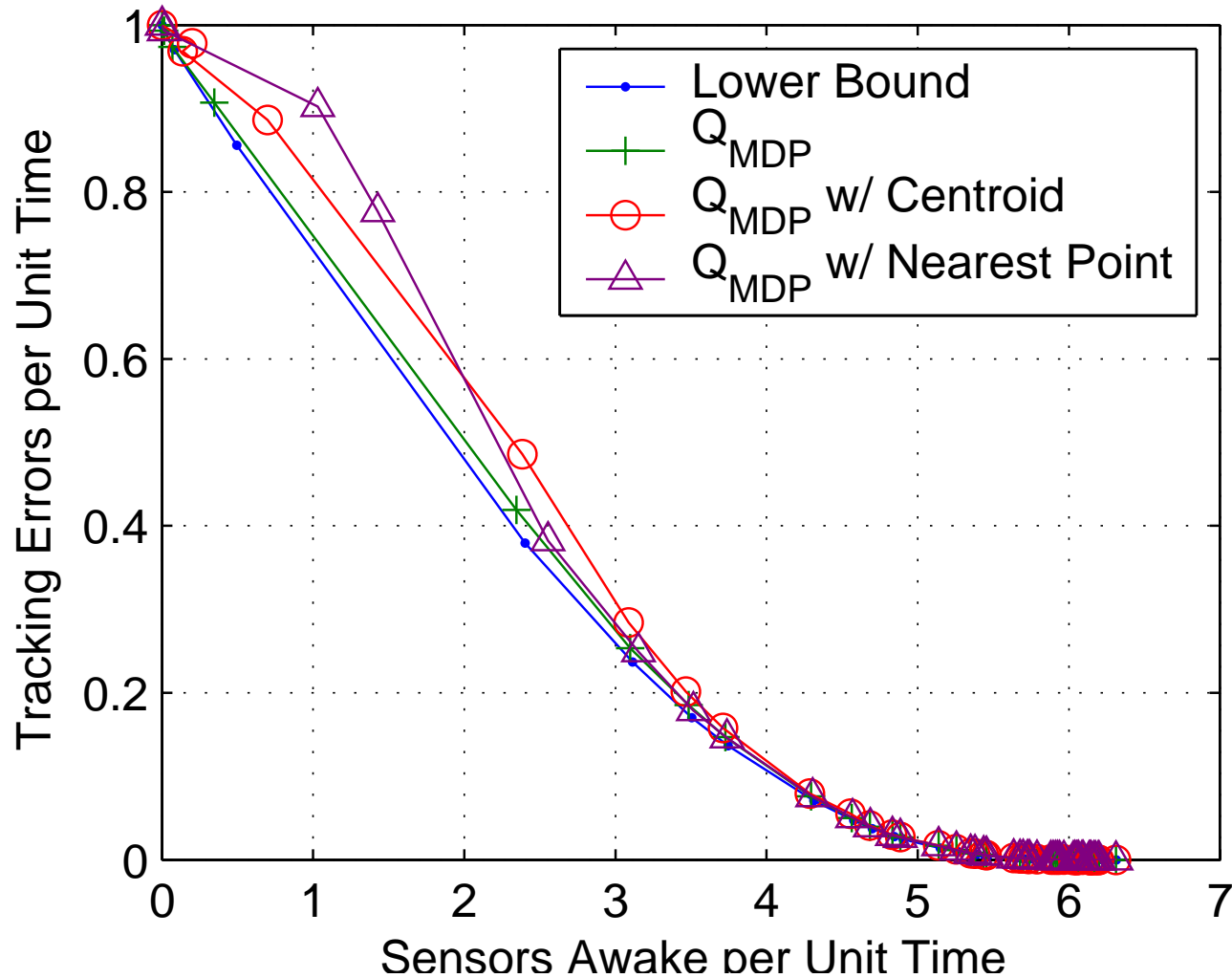
- Off-line computation also allows for decentralized implementation!



Results for Symmetric Random Walk

41 sensors, w_k i.i.d. Bernoulli(0.5) on $\{-1, +1\}$

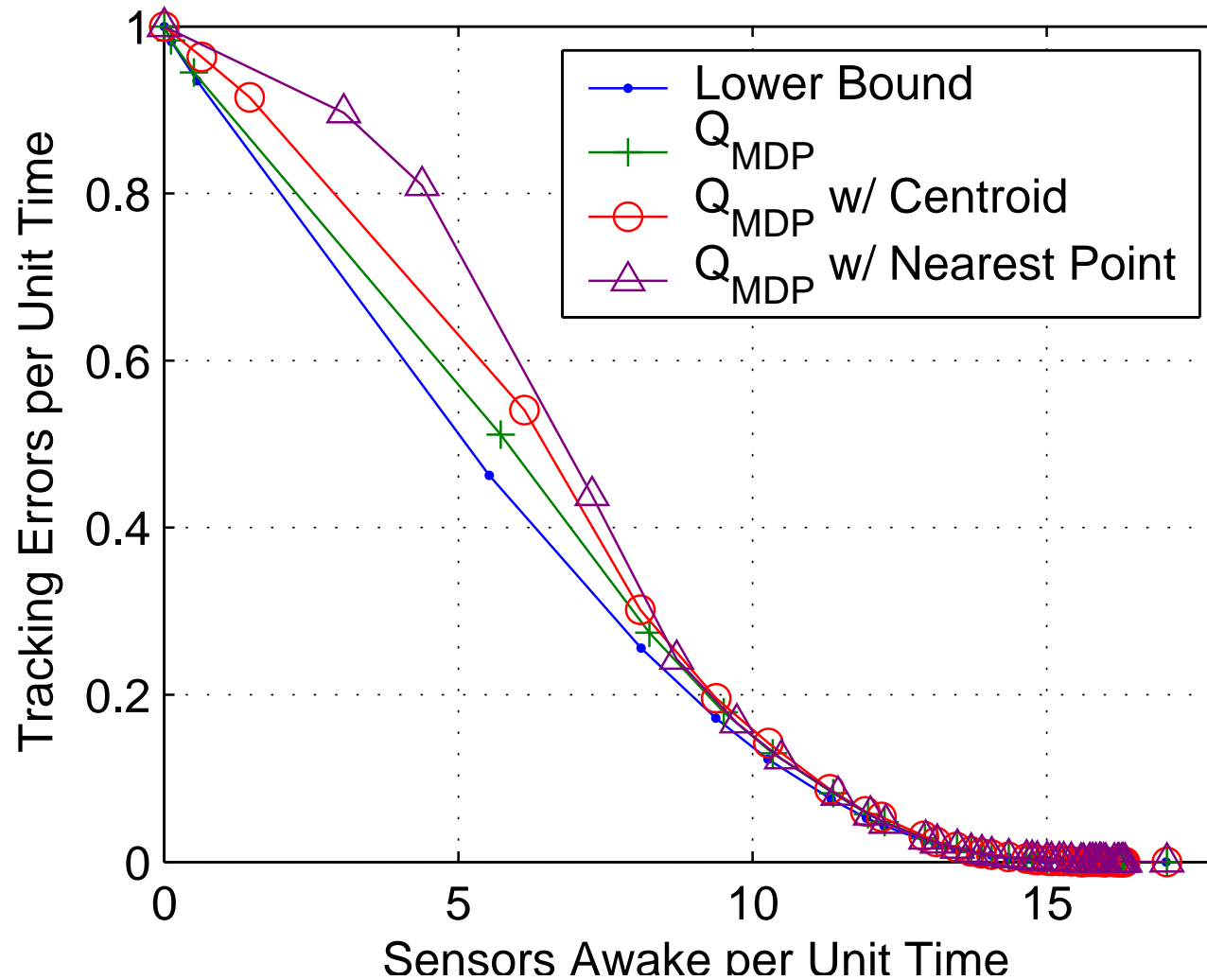
$m=20$, Symmetric Random Walk



Results for Three-Step Random Walk

61 sensors, w_k i.i.d. Uniform on $\{-3, -2, -1, 0, 1, 2, 3\}$

$m=30, \pm 3$ Max Uniform



Conclusions

- Tradeoff between energy consumption and tracking errors considerably improved by using object location information
- Optimal solution to tradeoff problem is intractable, but good suboptimal solutions can be designed
- **Current/Future work**
 - Decentralized implementation
 - Unknown statistics for random walk
 - Continuous tracking problem and selection of sampling time
 - Tracking multiple objects simultaneously
 - More complex sensing models
- Methodology can be applied to other sensing applications, e.g., process monitoring, change detection, etc.