Sensor Health State Estimation for Target Tracking with Binary Sensor Networks

Christos Laoudias, Michalis Michaelides and Christos Panayiotou

KIOS Research Center for Intelligent Systems and Networks
Department of Electrical and Computer Engineering
University of Cyprus, Nicosia, Cyprus

ΔΙΑΡΘΡΩΤΙΚΑ ΤΑΜΕΙΑ ΕΥΡΩΠΑΪΚΗ ΕΝΩΣΗ ΚΥΠΡΙΑΚΗ ΔΗΜΟΚΡΑΤΙΑ Research Promotion Foundation ΔΕΣΜΗ 2009-2010 University of Cyprus

Supported by the Cyprus Research Promotion Foundation under Grant ΤΠΕ/ΟΡΙΖΟ/0609(ΒΕ)/06

Outline

Introduction

Fault Model

Tracking Architecture

Simulation Results

Conclusions

Motivation of our work

- Binary sensor networks
  - Popular for demanding and safety critical applications, e.g. large area monitoring, target tracking

Motivation of our work

- Binary sensor networks
  - Popular for demanding and safety critical applications, e.g. large area monitoring, target tracking
- Living with faults
  - Sensing can be tampered (accidentally or deliberately) and detection/estimation suffers from faulty sensors
  - Tracking accuracy can be severely degraded

Motivation of our work

- Binary sensor networks
  - Popular for demanding and safety critical applications, e.g. large area monitoring, target tracking
- Living with faults
  - Sensing can be tampered (accidentally or deliberately) and detection/estimation suffers from faulty sensors
  - Tracking accuracy can be severely degraded
- Faulty sensors should NOT be used
  - Localization algorithms typically use all sensor readings regardless of the actual sensor's state
  - Sensor states are usually unavailable or extremely hard to obtain in real WSN applications
  - **Sensor Health State Estimation**: Intelligently select (at least mostly) healthy sensors for target tracking

Binary Sensor Network Model

Assumptions

1. A set of static sensor nodes $\ell_n = (x_n, y_n), \;\; n = 1, \dots, N$
2. A source moving at steady speed $\ell_s(t) = (x_s(t), y_s(t))$
3. The source emits a continuous omnidirectional signal

$$z_n(t) = \frac{c}{1 + d_n(t)^\gamma} + w_n(t),$$

where $d_n(t) = ||\ell_n - \ell_s(t)||$.

Sensor Alarm Status

$$A_n(t) = \begin{cases} 0 & \text{if } z_n(t) < T \\ 1 & \text{if } z_n(t) \geq T \end{cases}$$

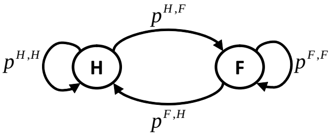
Sensor Fault Model

Stochastic Model

Markov Chain model with two discrete sensor states $s_n(t) \in \{F, H\}$

$$\pi_n(t+1) = C^T \pi_n(t)$$

- Sensor state probabilities $\pi_n(t) = [\pi_n^F(t)\ \pi_n^H(t)]^T$, $\pi_n^i(t) = \mathbf{P}[s_n(t) = i],\ i \in \{F, H\}$
- $C = \begin{bmatrix} p^{F,F} & p^{F,H} \\ p^{H,F} & p^{H,H} \end{bmatrix}$
- Steady state probabilities $\pi_n^i = \lim_{t\to\infty} \mathbf{P}[s_n(t) = i],\ i \in \{F, H\}$



Fault Generation

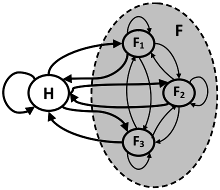
- Diverse fault types
- Different duration, e.g. temporary, permanent
- $p^{H,H} = 0.925$ and $p^{F,F} = 0.7$ gives $[\pi_n^F\ \pi_n^H]^T = [0.2\ 0.8]^T$
- $p^{F,F} = 1$ injects permanent faults

Sensor Fault Model

Stochastic Model

Markov Chain model with two discrete sensor states $s_n(t) \in \{F, H\}$

$$\pi_n(t+1) = C^T \pi_n(t)$$



- Sensor state probabilities $\pi_n(t) = [\pi_n^F(t)\ \pi_n^H(t)]^T$, $\pi_n^i(t) = \mathbf{P}[s_n(t) = i],\ i \in \{F, H\}$
- $C = \begin{bmatrix} p^{F,F} & p^{F,H} \\ p^{H,F} & p^{H,H} \end{bmatrix}$
- Steady state probabilities $\pi_n^i = \lim_{t\to\infty} \mathbf{P}[s_n(t) = i],\ i \in \{F, H\}$

Fault Generation

- Diverse fault types
- Different duration, e.g. temporary, permanent
- $p^{H,H} = 0.925$ and $p^{F,F} = 0.7$ gives $[\pi_n^F\ \pi_n^H]^T = [0.2\ 0.8]^T$
- $p^{F,F} = 1$ injects permanent faults

Types of faults

Reverse Status (RS)

- Sensors report the opposite readings than the expected ones
- Software bugs, compromised sensors, malicious network

Types of faults

Reverse Status (RS)

- Sensors report the opposite readings than the expected ones
- Software bugs, compromised sensors, malicious network

Stuck-At-1 (SA1)

- Sensors constantly report the presence of a source
- Board overheating, low battery, wrongly programmed threshold (i.e., low T), deployment of small decoy sources

Types of faults

Reverse Status (RS)

- Sensors report the opposite readings than the expected ones
- Software bugs, compromised sensors, malicious network

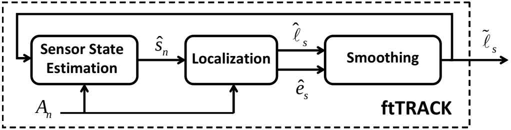
Stuck-At-1 (SA1)

- Sensors constantly report the presence of a source
- Board overheating, low battery, wrongly programmed threshold (i.e., low T), deployment of small decoy sources

Stuck-At-0 (SA0)

- Sensors fail to detect the source inside their ROC_n
- Dropped packets, high threshold T

Fault Tolerant Target Tracking Architecture



Sensor State Estimation component

- $\hat{s}_n(t)$: Estimated health state of each sensor

Localization component

- $\hat{\ell}_s(t)$: estimated target location
- $\hat{e}_s(t)$: estimation of the localization error (uncertainty)

Smoothing component

- $\tilde{\ell}_s(t)$: final location estimate (more accurate)

Overview of SNAP Localization

Subtract on Negative Add on Positive (SNAP) algorithm

- Event detection in binary sensor networks
- Low computational complexity and fault tolerance

Overview of SNAP Localization

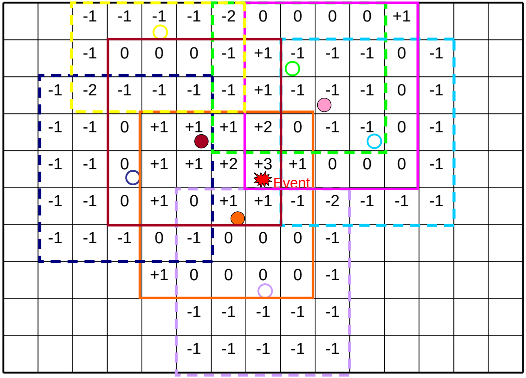
Subtract on Negative Add on Positive (SNAP) algorithm

- Event detection in binary sensor networks
- Low computational complexity and fault tolerance

Algorithm Steps

1. *Grid Formation:* The entire area is divided into a grid $\mathcal{G}$ with dimensions $R_x \times R_y$ and grid resolution g.
2. *Region of Coverage (ROC):* Given $\mathcal{G}$, the ROC_n of a sensor is a neighborhood of grid cells around the sensor node location.
3. *Likelihood Matrix $\mathcal{L}$ Construction:* All sensors add $+1$ (alarmed) or -1 (non-alarmed) to the cells that correspond to their ROC and contributions are added for each cell.
4. *Maximization:* The maximum value in $\mathcal{L}$ matrix, denoted as L_{max}, points to the estimated source location.

Example Application of SNAP



- Square ROC_n for alarmed and non-alarmed sensors
- Source is correctly localized in the grid cell with $L_{max} = +3$

Particle Filter Tracking

Target state and measurement model

$$X(t) = \Phi X(t-1) + \Gamma W(t-1) \quad (1)$$

$$Y(t) = MX(t) + U(t), \quad (2)$$

where $X(t) = [x_s(t) \; y_s(t) \; u_x(t) \; u_y(t)]^T$ is the target state

Particle Filter Steps

A set of particles $\{X^i(t-1)\}_{i=1}^{N_p}$ with weights $\{\omega^i(t-1)\}_{i=1}^{N_p}$

1. $X^i(t) = \Phi X^i(t-1) + \Gamma W(t-1)$
2. $\left(\hat{\ell}_s(t), \hat{e}_s(t)\right) = SNAP(\hat{s}_n(t), A_n(t))$
3. $\omega^i(t) = \omega^i(t-1)p(t), \;\; p(t) = \frac{1}{\sqrt{2\pi}\sigma(t)} \exp(-\frac{(\bar{X}^i(t)-\hat{\ell}_s(t))^2}{2\sigma(t)^2})$
4. $\omega^i(t) = \omega^i(t) / \sum_{i=1}^{N_p} \omega^i(t)$ and Linear Time Resampling
5. $\tilde{\ell}_s(t) = \sum_{i=1}^{N_p} \omega^i(t) X^i(t)$

ML Sensor State Estimation

The estimator is based on a Markov Chain model

$$\hat{\pi}_n(t+1) = \hat{C}_n(t)^T \hat{\pi}_n(t), \tag{3}$$

where $\hat{\pi}_n(t) = [\hat{\pi}_n^F(t)\ \hat{\pi}_n^H(t)]^T$, $\hat{\pi}_n^i(t) = \mathbf{P}[\hat{s}_n(t) = i]$, $i \in \{F, H\}$

$$\hat{C}_n(t) = \begin{bmatrix} \hat{p}_n^{F,F}(t) & \hat{p}_n^{F,H}(t) \\ \hat{p}_n^{H,F}(t) & \hat{p}_n^{H,H}(t) \end{bmatrix}, \tag{4}$$

where $\hat{p}_n^{i,j}(t) \neq p^{i,j}$ $i, j \in \{F, H\}$.

Binary error signal $r_n(t)$

$$r_n(t) = \begin{cases} 1 & \text{if } d_n(t) \leq R_I \text{ AND } A_n(t) = 0 \\ 1 & \text{if } d_n(t) > R_I \text{ AND } A_n(t) = 1 \\ 0 & \text{if } d_n(t) \leq R_I \text{ AND } A_n(t) = 1 \\ 0 & \text{if } d_n(t) > R_I \text{ AND } A_n(t) = 0 \end{cases} \tag{5}$$

ML Sensor State Estimation

Main Idea

Obtain $\hat{s}_n(t+1)$ by calculating the probability of a sensor being at a specific state given the current error signal, i.e. $\hat{\pi}_n^{i|q}(t) = \mathbf{P}[s_n(t) = i | r_n(t) = q],\ i \in \{F, H\},\ q \in \{0, 1\}$.

ML Sensor State Estimate

$$\hat{s}_n(t+1)|_{r_n(t)=q} = \arg \max_{i \in \{F,H\}} \hat{\pi}_n^{i|q}(t),\ q \in \{0,1\}. \tag{6}$$

Using Bayes' rule

$$\hat{\pi}_n^{i|q}(t) = \frac{\mathbf{P}[r_n(t) = q | s_n(t) = i] \hat{\pi}_n^{i}(t)}{\mathbf{P}[r_n(t) = q]} \tag{7}$$

$$\hat{\pi}_n^{i|q}(t) = \frac{\mathbf{P}[r_n(t) = q | s_n(t) = i] \hat{\pi}_n^{i}(t)}{\sum_{j \in \{F,H\}} \mathbf{P}[r_n(t) = q | s_n(t) = j] \hat{\pi}_n^{j}(t)} \tag{8}$$

ML Sensor State Estimation

Definitions

Probability of a sensor having a *wrong* output given its state

- $p_n^h(t) = \mathbf{P}[r_n(t) = 1 | s_n(t) = H]$
- $p_n^f(t) = \mathbf{P}[r_n(t) = 1 | s_n(t) = F]$

$$\hat{\pi}_n^{F|1}(t) = \frac{p_n^f(t) \cdot \hat{\pi}_n^F(t)}{p_n^f(t) \cdot \hat{\pi}_n^F(t) + p_n^h(t) \cdot \hat{\pi}_n^H(t)} \tag{9}$$

$$\hat{\pi}_n^{H|1}(t) = \frac{p_n^h(t) \cdot \hat{\pi}_n^H(t)}{p_n^f(t) \cdot \hat{\pi}_n^F(t) + p_n^h(t) \cdot \hat{\pi}_n^H(t)} \tag{10}$$

$$\hat{\pi}_n^{F|0}(t) = \frac{(1 - p_n^f(t)) \cdot \hat{\pi}_n^F(t)}{(1 - p_n^f(t)) \cdot \hat{\pi}_n^F(t) + (1 - p_n^h(t)) \cdot \hat{\pi}_n^H(t)} \tag{11}$$

$$\hat{\pi}_n^{H|0}(t) = \frac{(1 - p_n^h(t)) \cdot \hat{\pi}_n^H(t)}{(1 - p_n^f(t)) \cdot \hat{\pi}_n^F(t) + (1 - p_n^h(t)) \cdot \hat{\pi}_n^H(t)}. \tag{12}$$

ML Sensor State Estimation

In case the sensor output is *wrong*, i.e. $r_n(t) = 1$

$$\hat{s}_n(t+1)|_{r_n(t)=1} = \begin{cases} H & \text{if } \hat{\pi}_n^H(t) > \frac{p_n^f(t)}{p_n^f(t)+p_n^h(t)} \\ F & \text{otherwise} \end{cases} \tag{13}$$

In case the sensor output is *correct*, i.e. $r_n(t) = 0$

$$\hat{s}_n(t+1)|_{r_n(t)=0} = \begin{cases} F & \text{if } \hat{\pi}_n^H(t) < \frac{1-p_n^f(t)}{2-p_n^f(t)-p_n^h(t)} \\ H & \text{otherwise} \end{cases} \tag{14}$$

- Only $\hat{\pi}_n^H(t)$, $p_n^h(t)$ and $p_n^f(t)$ need to be computed for estimating the sensor health state, given that $r_n(t)$ is known
- **Problem:** $r_n(t)$ is not available (target location is unknown)
- **Solution:** $\tilde{r}_n(t)$ estimates $r_n(t)$ by substituting $d_n(t)$ with $\tilde{d}_n(t)$, where $\tilde{d}_n(t) = ||\ell_n - \tilde{\ell}_s(t)||$

Simple Estimator

Assumption

The error signal $\tilde{r}_n(t)$ is always equal to 1 when the sensor is *Faulty* and always equal to 0 when the sensor is *Healthy*.

Sensor State Estimate

This means that $p_n^f(t) = 1$ and $p_n^h(t) = 0, \ \forall t$ leading to

$$\hat{s}_n(t+1) = \begin{cases} H & \text{if } \tilde{r}_n(t) = 0 \\ F & \text{if } \tilde{r}_n(t) = 1 \end{cases} \quad (15)$$

- **Intuition:** If we fully trust the error signal, then the sensor health state is reliably estimated by $\tilde{r}_n(t)$
- **Problem:** Fully trusting the error signal $\tilde{r}_n(t)$ is not a good strategy
- **Solution:** Incorporate previous estimations that are encapsulated in the estimated sensor state probabilities

Static Estimator

Assumption

The Markov Chain in the Sensor State Estimator has reached equilibrium.

Sensor State Estimate

We may employ an estimate of the unknown steady state probability $\hat{\pi}_n^H$ to determine the sensor health state as

$$\hat{s}_n(t+1)|_{r_n(t)=1} = \begin{cases} H & \text{if } \hat{\pi}_n^H > \frac{p_n^f(t)}{p_n^f(t)+p_n^h(t)} \\ F & \text{otherwise} \end{cases} \tag{16}$$

$$\hat{s}_n(t+1)|_{r_n(t)=0} = \begin{cases} F & \text{if } \hat{\pi}_n^H < \frac{1-p_n^f(t)}{2-p_n^f(t)-p_n^h(t)} \\ H & \text{otherwise} \end{cases} \tag{17}$$

Static Estimator

The steady state probabilities are computed with

$$\begin{bmatrix} \hat{\pi}_n^F \\ \hat{\pi}_n^H \end{bmatrix} = \hat{C}_n^T(t) \begin{bmatrix} \hat{\pi}_n^F \\ \hat{\pi}_n^H \end{bmatrix}, \tag{18}$$

where $\hat{p}_n^{i,j}(t)$ in $\hat{C}_n(t)$ can be estimated online by

$$\hat{p}_n^{i,j}(t) = \frac{R_n^{i,j}(t)}{\sum_{k\in\{F,H\}} R_n^{i,k}(t)}, \quad i,j \in \{F,H\}, \tag{19}$$

where $R_n^{i,j}(t)$ increases by one if $\hat{s}_n(t-1) = i$ and $\hat{s}_n(t) = j$.

Calculation of $p_n^h(t)$ and $p_n^f(t)$

$$p_n^h(t) = \big(1 - Q_w(t)\big)\big(1 - Q_d(t)\big) + Q_w(t)Q_d(t) \tag{20}$$

$$p_n^f(t) = \big(1 - Q_w(t)\big)Q_d(t) + Q_w(t)\big(1 - Q_d(t)\big) \tag{21}$$

$$Q_w(t) = Q\left(\frac{T-\mu_n(t)}{\sigma_w}\right),\ \mu_n(t) = \frac{c}{1+\tilde{d}_n(t)^\gamma},\ Q_d(t) = Q\left(\frac{R_l-\tilde{d}_n(t)}{\sigma_d}\right).$$

Dynamic Estimator

Main Idea

Consider the error signal not only for estimating the unknown sensor state, but also for updating the estimated sensor state probabilities.

$$\begin{bmatrix} \hat{\pi}_n^F(t+1) \\ \hat{\pi}_n^H(t+1) \end{bmatrix} = \hat{C}_n^T(t) \begin{bmatrix} \hat{\pi}_n^{F|q}(t) \\ \hat{\pi}_n^{H|q}(t) \end{bmatrix}, \quad q \in \{0, 1\} \tag{22}$$

- **Intuition:** All previous observations of the error signal are encapsulated in the estimated sensor state probabilities, thus affecting the future estimation steps.

Simulation Setup

Sensor field

100×100 field, $N = 600$ sensors, single source, staircase path
$M = 180$

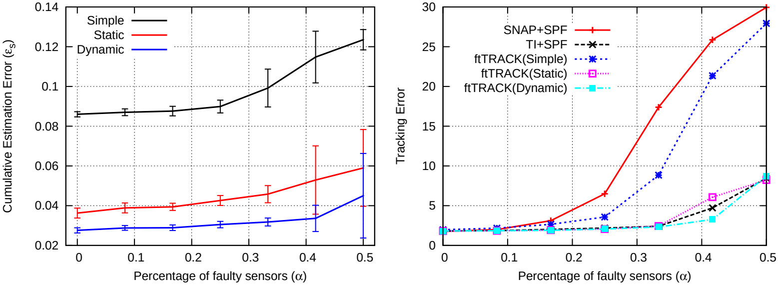
Fault model

2-state Markov Chain with varying $p^{i,j}$, $i, j \in \{F, H\}$ to generate temporary and permanent faults

Performance Metrics

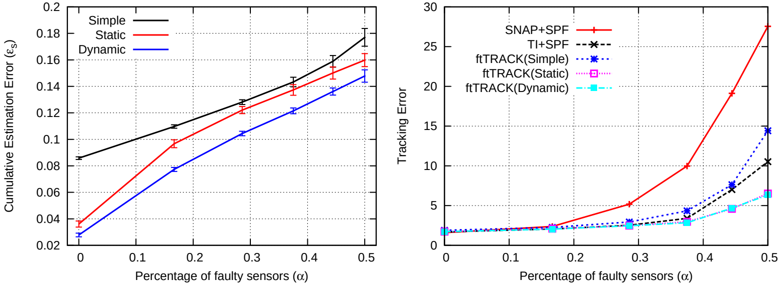
- Cumulative state estimation error $\mathcal{E}_s = \frac{1}{NM} \sum_{t=1}^{M} \sum_{n=1}^{N} \epsilon_n(t)$
  - $\epsilon_n(t) = \begin{cases} 0 & \text{if } \hat{s}_n(t) = s_n(t) \\ 1 & \text{if } \hat{s}_n(t) \neq s_n(t) \end{cases}$
- Tracking error $E_T = \frac{1}{M} \sum_{t=1}^{M} ||\tilde{\ell}_s(t) - \ell_s(t)||$

Results (Permanent faults)



- $z_n(t) = \frac{5000}{1+d_n(t)^2} + w_n(t),\ w_n \sim \mathcal{N}(0, 1000),\ T = 50$ and $R_I = 10$
- Reverse Status faults
- Adaptive particle filter with $N_p = 500$ particles

Results (Permanent and Temporary faults)



- Temporary mixed and permanent Reverse Status faults

Concluding Remarks

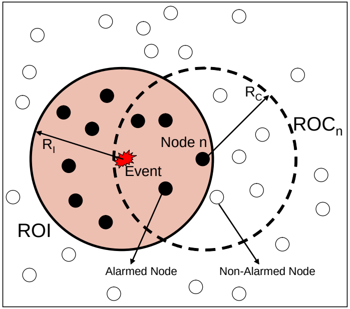
- Introduced a Markov Chain fault model to generate different types of real faults documented in the literature
- The proposed architecture addresses the joint target tracking and sensor health state estimation problem in binary WSNs
- Maintain a high level of tracking accuracy, even when a large number of sensors in the field fail
- Next steps
  - Incorporate the correlation of the alarm status $A_n(t)$ for neighboring sensors into the error signal $r_n(t)$
  - Decentralized architecture for multiple target tracking

Thank you for your attention

Contact
Christos Laoudias
KIOS Research Center for Intelligent Systems and Networks
Department of Electrical & Computer Engineering
University of Cyprus
Email: laoudias@ucy.ac.cy

Extra Slides

Regions of Interest



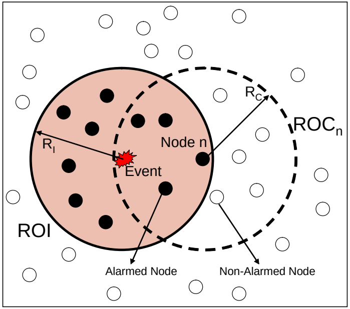
Region of Influence (ROI)

Area around the source where a sensor is alarmed with $p \geq 0.5$

Region of Coverage (ROC_n)

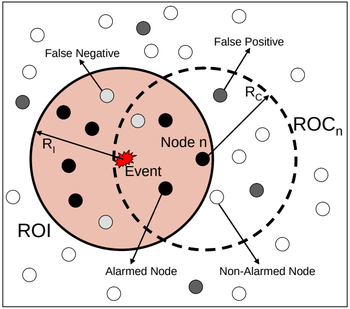
Area around a sensor n where a source (if present) it will be detected with $p \geq 0.5$

Erroneous Sensor Behaviour



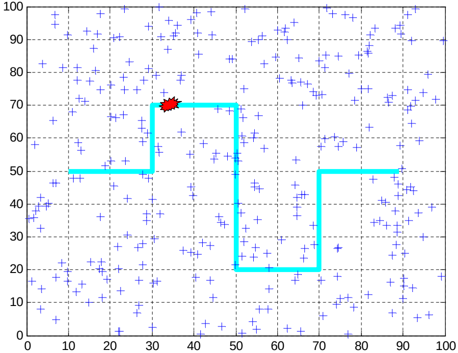
- *False Positive* and *False Negative* sensors

Erroneous Sensor Behaviour

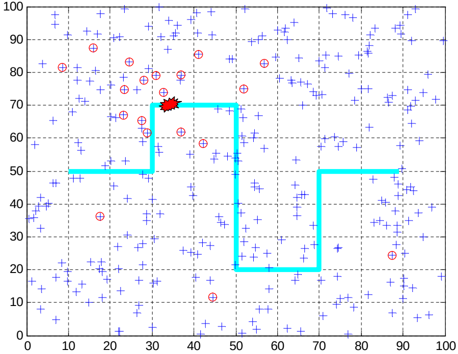


- *False Positive* and *False Negative* sensors

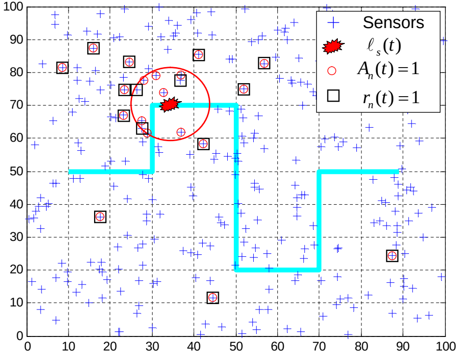
Interpretation of the Error Signal



Interpretation of the Error Signal

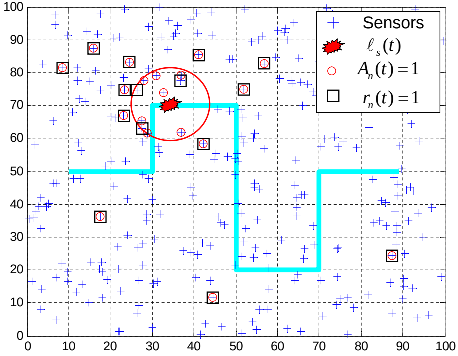


Interpretation of the Error Signal



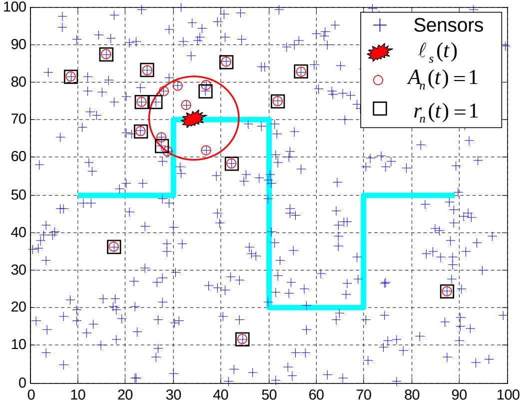
- $r_n(t) = 1$ (sensor output is *wrong*)
  - sensor n is inside the *ROI* and is non-alarmed or
  - sensor n is outside the *ROI* and is alarmed

Interpretation of the Error Signal

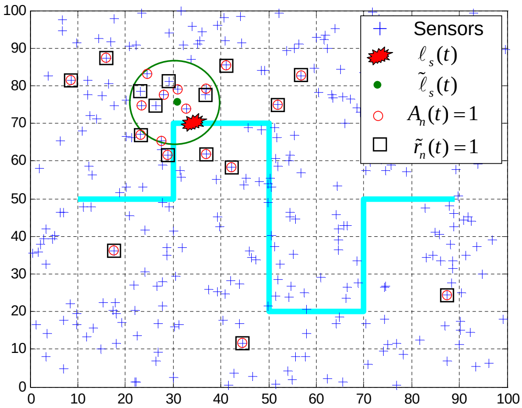


- $r_n(t) = 1$ (sensor output is *wrong*)
  - sensor n is inside the *ROI* and is non-alarmed or
  - sensor n is outside the *ROI* and is alarmed
- $r_n(t) = 0$ (sensor output is *correct*)
  - sensor n is inside the *ROI* and is alarmed or
  - sensor n is outside the *ROI* and is non-alarmed

$r_n(t)$ vs $\tilde{r}_n(t)$

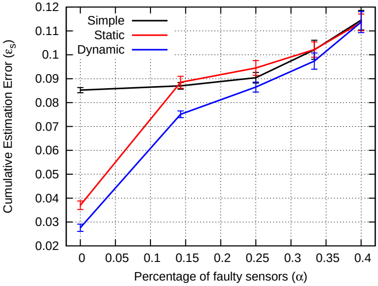


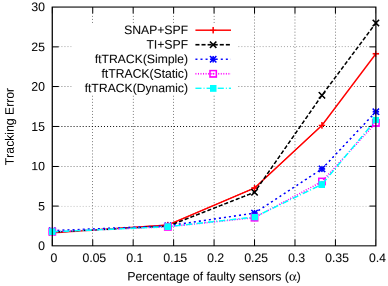
$r_n(t)$ vs $\tilde{r}_n(t)$



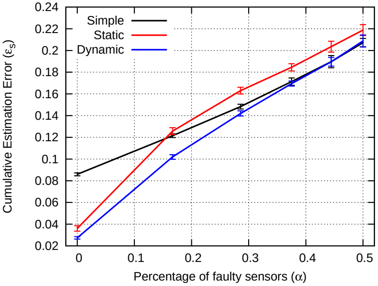
- In this scenario $\tilde{r}_n(t) \neq r_n(t)$ for 6 sensors

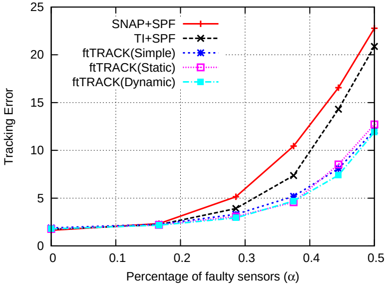
Results with SNAP (RS faults)





Results (RS and SA faults)





Results with SNAP (SA faults)

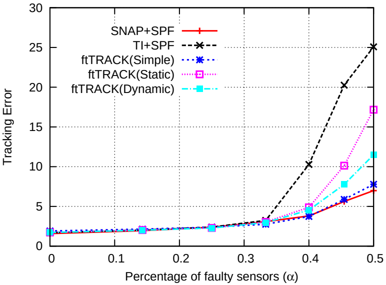


Figure: SA1 faults.

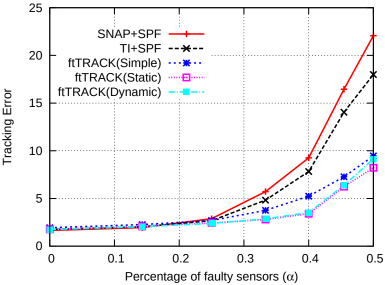


Figure: SA0 faults.

Results with Variable Source Energy

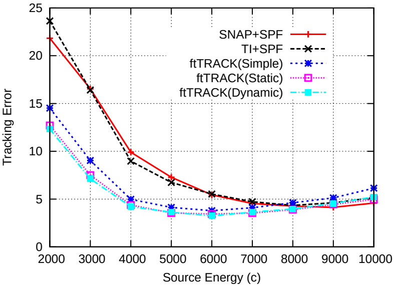


Figure: Temporary RS faults ($\alpha = 25\%$).

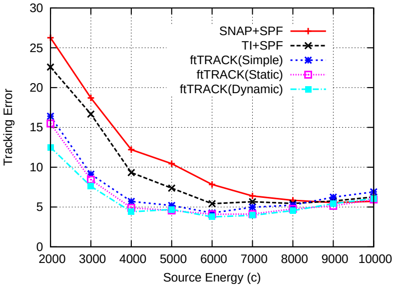
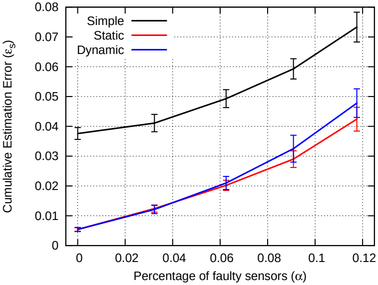
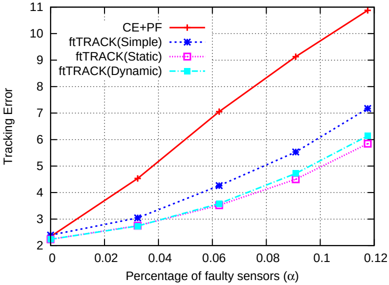


Figure: Temporary mixed faults ($\alpha = 38\%$).

Results with CE (RS faults)





- $z_n(t) = \frac{3000}{1+d_n(t)^2} + w_n(t),\ w_n \sim \mathcal{N}(0,1),\ T = 5$ and $R_I = 24.5$
- Centroid Estimator $\hat{\ell}_s(t) = \left(\frac{1}{P}\sum_{p=1}^{P} x_p, \frac{1}{P}\sum_{p=1}^{P} y_p\right)$
  - $(x_p, y_p),\ p = 1, \ldots, P\ (P \leq N)$ and $A_p(t) = 1$
- Standard particle filter with $N_p = 500$ particles

Results with CE (RS and SA faults)