



MoteTrack: A Robust, Decentralized Approach to RF-Based Location Tracking

Konrad Lorincz and Matt Welsh

Harvard University
Division of Engineering and Applied Sciences
{konrad, mdw}@eecs.harvard.edu

September 10, 2004



Why Location Tracking is Useful

Sensor networks are about “observing events” – for many applications the physical location of the event is important!

Our Primary Focus: Disaster response applications where the robustness of the location-tracking infrastructure is at stake

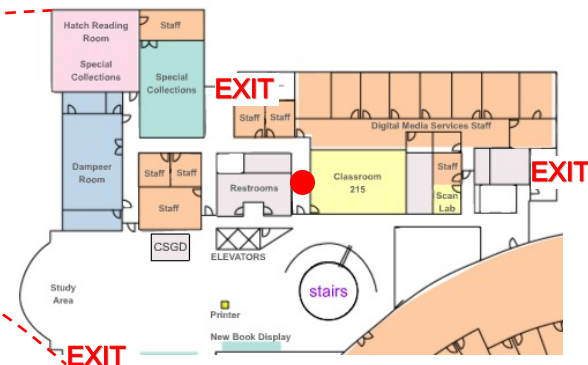
Why Location Tracking is Useful

Firefighters entering a burning building

- Often cannot see because of heavy smoke + are unfamiliar with building
- Can greatly benefit from a heads-up display to track their location and monitor safe exit routes [McKinsey & Co. reports, Aug. 19, 2002]
 - Chicago City Council ... all buildings more than 80 feet tall must submit electronic floor plans [Forefront, Fall 2003]
- Incident commander can better coordinate rescuers from command post



developed at BMI

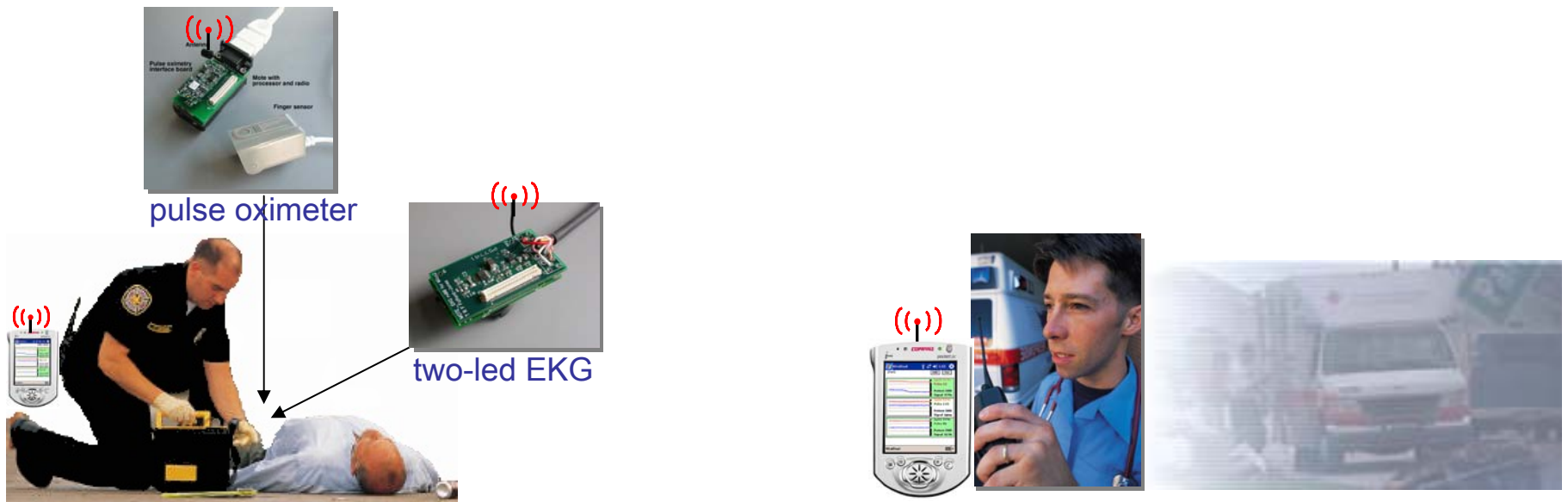


heads-up display

Why Location Tracking is Useful

Emergency Medical Care

- Monitor the location of patients with wearable vital sign sensors
 - Can quickly locate patient that suddenly requires immediate attention (e.g., patient stops breathing)
- Monitoring the location of emergency personnel (e.g., EMTs, firefighters)
 - 6 firefighters die in Worcester MA fire, [Boston Globe, Dec. 10, 1999]



PDA-based triage app.



Outline

- Motivation
- **Introduction**
- System Description
- Implementation and Data Collection
- Evaluation
- Future Work and Directions
- Conclusions



Requirements and Problems

Requirements for safety-critical applications

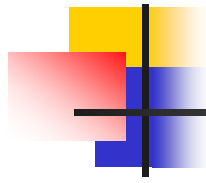
- Robustness
 - to beacon node failure
 - to protocol for calculating locations
 - to incomplete data and RF perturbations
- Must run on small, low-power battery operated devices that can be readily embedded into equipment or the environment
- Reasonable accuracy (a couple of meters)

Lots of prior work:

[RADAR, WLAN, LOCADIO, SpotOn, PlaceLab, ...]

but are inappropriate for these applications because of one or more of the following:

- Centralized
- Require powered/wired infrastructure
- Not robust to node failures, incomplete data, or RF perturbations
- Accuracy too coarse grained



Achieving Robustness

Our Approach

- Base the location tracking on empirical measurements of radio signals from multiple transmitters, using an algorithm similar to RADAR
- Achieve robustness by extending this approach in 3 significant ways
 - Use a decentralized approach for computing locations that runs on the beacon nodes, rather than a central server
 - Replicate the location signature database across the beacon nodes in a fashion that minimizes per-node storage overhead and achieves high robustness to failure
 - Employ a signature distance metric that adapts to partial failure of beacon nodes, partial information, and perturbations in the RF signal

Platform Choice

Why Motes/TinyOS seems to be the right platform

- Small size
 - Easy to embed in environment and equipment
- Can operate off of battery + it is low power
 - Resilient to infrastructure failure
- Well established platform
 - Used by over 150 research groups worldwide
 - Easy to integrate new sensors/actuators

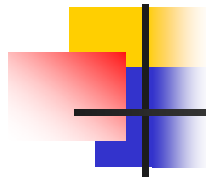


Mica2 mote



Mica2 Dot mote

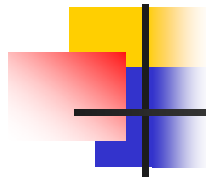




Why RF-based

Why RF-based (Radio Frequency)

- It's the “least common denominator” for most wireless sensor networks
 - Can supplement other wireless sensor networks with location tracking for little or no additional cost
- Benefits of no additional hardware
 - Lower cost and smaller size
- “Robust” to the environment
 - Doesn't require line of sight
 - Is not sensitive to variations of light and temperature



Challenges

Accurate RF-based localization with low-power radios

- Large fluctuations in received signal strength indicator (RSSI)
- No clear correlation with distance

Low device capability and memory size

- e.g., Mica2: 4KB RAM, 8-bit CPU, no floating point unit
- Can't do real signal processing

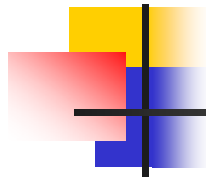
Robustness

- Failed nodes
- RF characteristics of building changed
- Incomplete data



Outline

- Motivation
- Introduction
- **System Description**
- Implementation and Data Collection
- Evaluation
- Future Work and Directions
- Conclusions



High-level Overview of MoteTrack

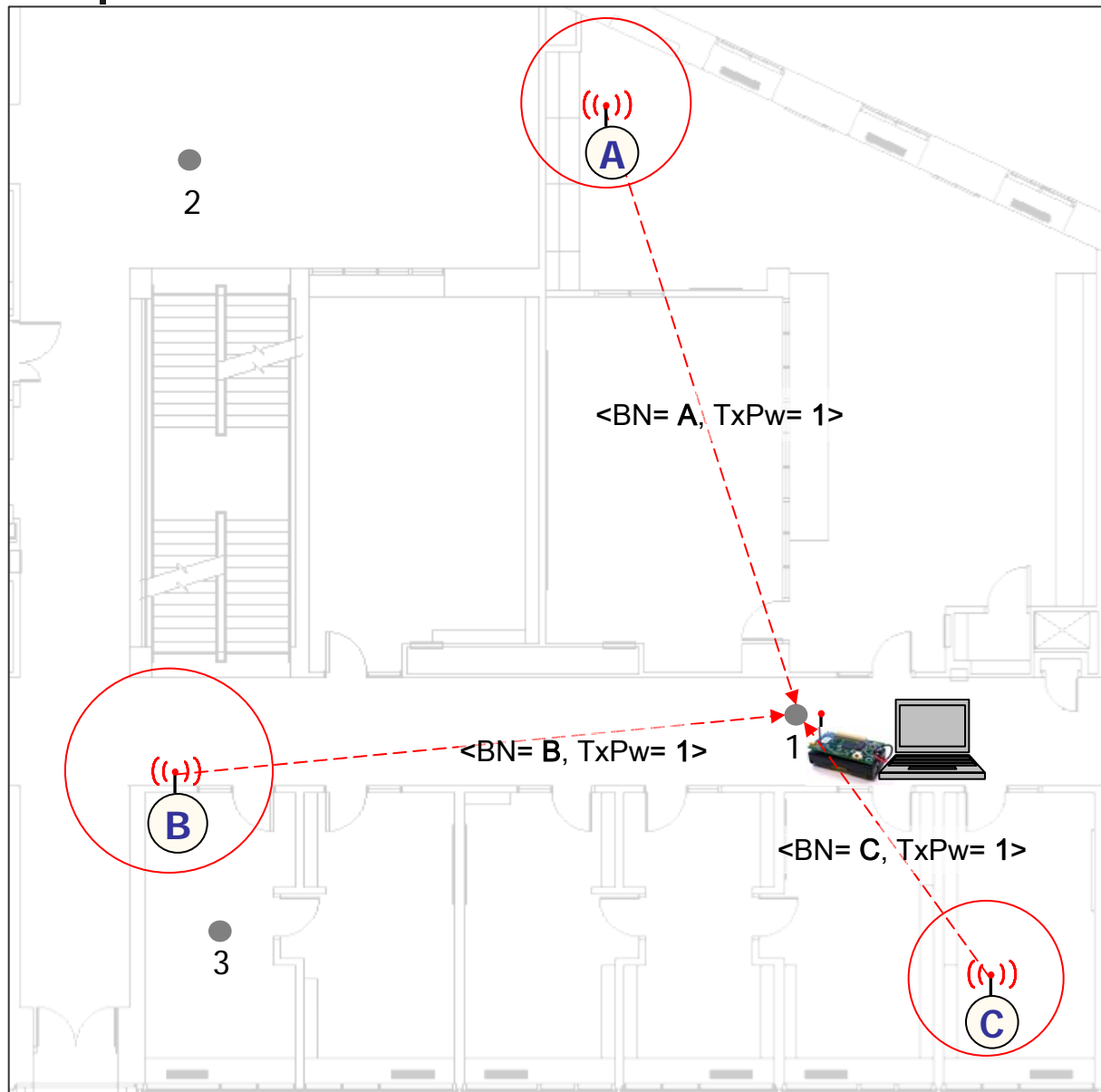
Basic premise of MoteTrack:

- Use empirical results of signal strength obtained from beacon nodes to determine distance.

MoteTrack consists of two phases:

- Phase I: (initial setup – performed once)
 - Placement of *beacon nodes* at various locations in the environment
 - Construction of a “signal strength map” or *reference signature database*, by going around the environment at known locations and collecting samples
- Phase II: (location estimation – normal operation)
 - Estimate the location of *mobile nodes*

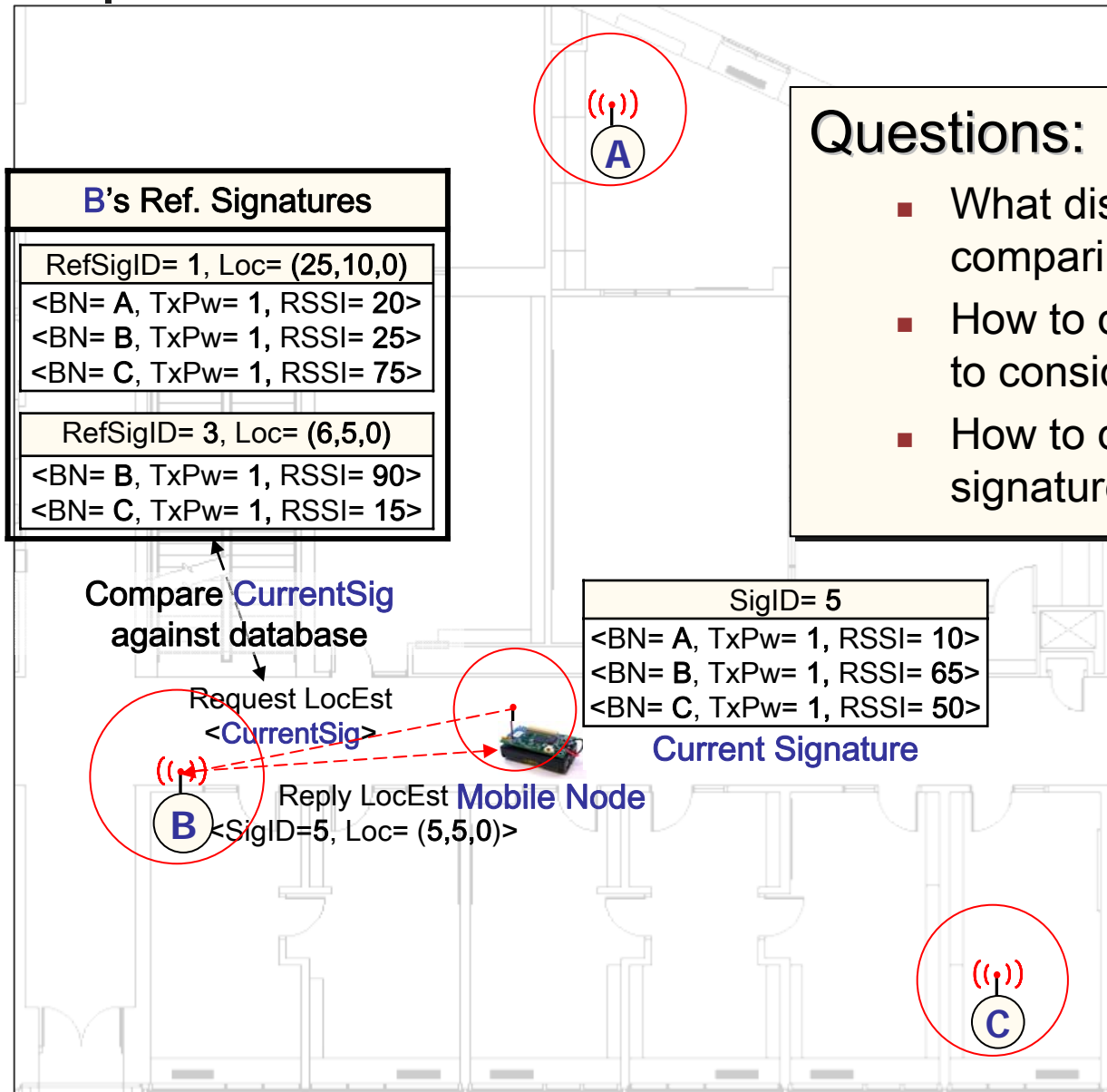
Phase I: Initial Setup



Reference Signature Database
RefSigID= 1, Loc= (25,10,0)
$\langle \text{BN} = \text{A}, \text{TxPw} = 1, \text{RSSI} = 20 \rangle$
$\langle \text{BN} = \text{B}, \text{TxPw} = 1, \text{RSSI} = 25 \rangle$
$\langle \text{BN} = \text{C}, \text{TxPw} = 1, \text{RSSI} = 75 \rangle$
RefSigID= 2, Loc= (5,30,0)
$\langle \text{BN} = \text{A}, \text{TxPw} = 1, \text{RSSI} = 80 \rangle$
RefSigID= 3, Loc= (6,5,0)
$\langle \text{BN} = \text{B}, \text{TxPw} = 1, \text{RSSI} = 90 \rangle$
$\langle \text{BN} = \text{C}, \text{TxPw} = 1, \text{RSSI} = 15 \rangle$

- Beacon message sent at multiple TX powers (only one shown here for simplicity)

Phase II: Location Estimation



Questions:

- What distance metric to use for comparing two signatures?
- How to decide which ref. signatures to consider for estimating location?
- How to distribute/replicate the ref. signature db. to beacon nodes?

- Each beacon node stores a slice of the entire reference signature database



Distance Metric Between Signatures

Let

- s be the current signature
- R be a set of reference signatures, where $r \in R$
- T be the set of signature tuples in both r and s

Distance metric used: *Manhattan*

$$M(r, s) = \sum_{t \in T} | \text{meanRSSI}(t)_r - \text{meanRSSI}(t)_s |$$

Accounting for missing data

- In general the set of beacon nodes and TX power levels in signatures r and s will be different
- Reasons: measurements taken at *different locations* in the building or *beacon node failure*



Adaptive Signature Distance Metric

How to deal with missing data?

- *Bidirectional* comparison (appropriate with few failures)

$$M_{bidirectional}(r, s) = M(r, s) + \beta \sum_{T \in (s-r)} meanRSSI(t)_s + \beta \sum_{T \in (r-s)} meanRSSI(t)_r$$

- *Unidirectional* comparison (appropriate with many failures)

$$M_{unidirectional}(r, s) = M(r, s) + \beta \sum_{T \in (s-r)} meanRSSI(t)_s$$

Example: (with 1 TX power and $\beta = 1$)

<i>r</i>	<i>s</i>
BN 1, TxPw 1, RSSI 20	BN 1, TxPw 1, RSSI 45
	BN 2, TxPw 1, RSSI 15
BN 3, TxPw 1, RSSI 70	
BN 4, TxPw 1, RSSI 90	BN 4, TxPw 1, RSSI 60

$$M_{bidirectional} = |20 - 45| + |90 - 60| + 15 + 70$$

$$M_{unidirectional} = |20 - 45| + |90 - 60| + 15$$



Adaptive Signature Distance Metric

Adaptive approach

- Periodically estimate the *local failure ratio*
 - Take intersection between the current and original neighborhoods
- Decide dynamically which comparison algorithm to use

Assumptions

- Connectivity between nodes doesn't change substantially over time
- No beacon node failures between the time the reference signature database is collected and deployment



Selecting Reference Signatures

How to decide which reference signatures to consider?

- *k-nearest*

- Takes the centroid or weighted centroid of the k nearest reference signatures (in signal space)
 - *Pros:* simple and straightforward
 - *Cons:* doesn't account for non-uniform density of reference signatures ... may compare against very distant signatures

- *threshold-nearest*

- Instead of *fixed k* or *fixed distance*, it uses a threshold relative to the signature distance to the nearest reference signature

Includes all reference signatures r in R , that satisfy

$$r^* = \arg \min_{r \in R} M(r, s)$$

$$c > \frac{M(r, s)}{M(r^*, s)}$$

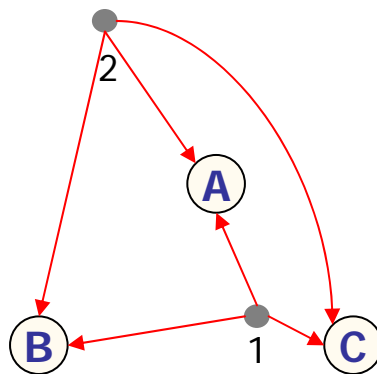
for a given threshold c

Distributing Reference Signature Database

Distributing/replicating reference signatures to beacon nodes

- *Greedy*

- Each beacon node stores the k closest reference signatures
 - *Pros:* Doesn't require global knowledge of ref. signatures or beacon nodes
 - *Cons:* Some ref. signatures may never get assigned, while other replicated a very large number of times!



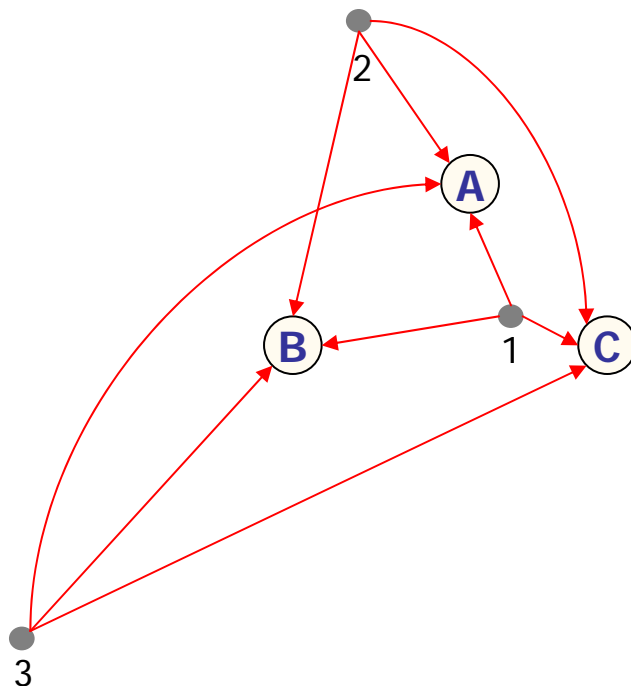
- for $k=1$, ref. signatures 2 & 3 don't get assigned to a beacon node
- for $k=2$, ref. signature 3 doesn't get assigned to a beacon node

Distributing Reference Signature Database

Distributing reference signatures to beacon nodes

- *Balanced*

- Ensures that each reference signature is replicated k times while trying to pair each beacon node with its closest reference signatures (inspired by the stable marriage algorithm)
 - *Pros:* Ensures all ref. sigs. are replicated k times
 - *Cons:* Requires global knowledge of all ref. sigs. and beacon nodes.



Algorithm (high-level)

repeat until all ref-sigs are paired k times

Pair next closest <ref-sig, BN> pair, that doesn't violate one of the following

1. no ref-sig is paired with a BN more than one additional time from any other ref-sig
2. no BN is paired with a ref-sig more than one additional time from any other BN

nbr rep	time steps →								
	$k = 1$			$k = 2$			$k = 3$		
A		2		1					3
B			3		2		1		
C	1					3		2	



Decentralized Loc. Estimation Protocols

k beacon nodes send their reference signature slice

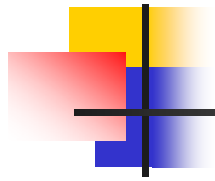
- Mobile node sends signature to the k closest beacon nodes and each beacon node replies with its reference signature slice
 - *Pros:* larger set of reference signatures; can be very accurate
 - *Cons:* high communication overhead

k beacon nodes send their location estimate

- Mobile node sends signature to the k closest beacon nodes and each beacon node replies with its location estimate. Mobile node takes centroid of received locations.
 - *Pros:* location estimates are much smaller than ref. signatures
 - *Cons:* not very accurate loc. est.

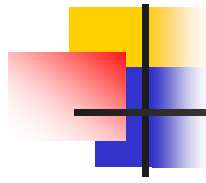
Max-RSSI beacon node sends its location estimate

- Mobile node sends signature to the closest beacon node, which replies with location estimate
 - *Pros:* very low communication overhead and accurate estimates
 - *Cons:* can be less accurate when the nbr. ref. sigs. per node is small



Outline

- Motivation
- Introduction
- System Description
- **Implementation and Data Collection**
- **Evaluation**
- Future Work and Directions
- Conclusions



Implementation and Data Collection

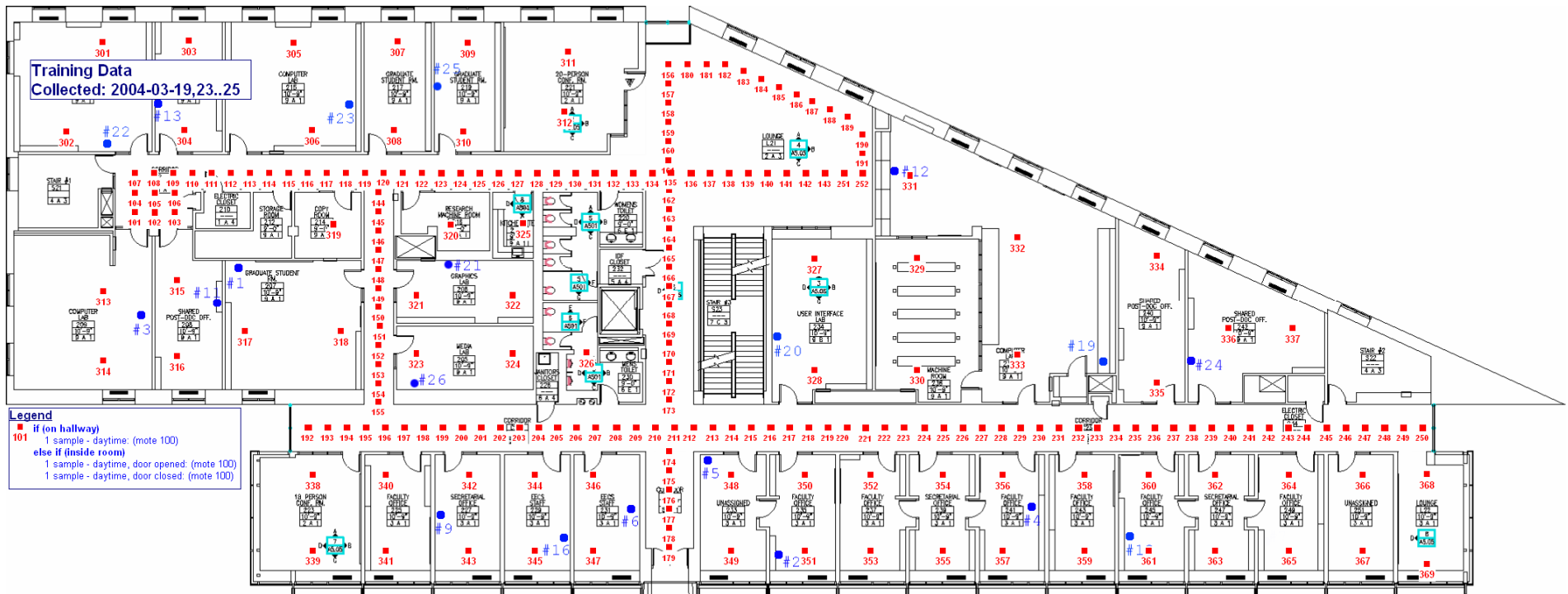
Implementation

- On Mica2 using TinyOS
 - 8-bit CPU, 4 KB RAM, 128 KB prog. mem., 433 MHz FSK transceiver
- Runs entirely on motes, except during the data collection phase
- ~ 4,100 lines of NesC code (beacon + mobile node) and ~ 6,000 lines of Java code
- Reference signatures on beacon nodes stored in program memory
 - 108 bytes per ref. sig. (at 3 TX powers); ~30KB for 282 ref. sigs.

Data Collection and Deployment

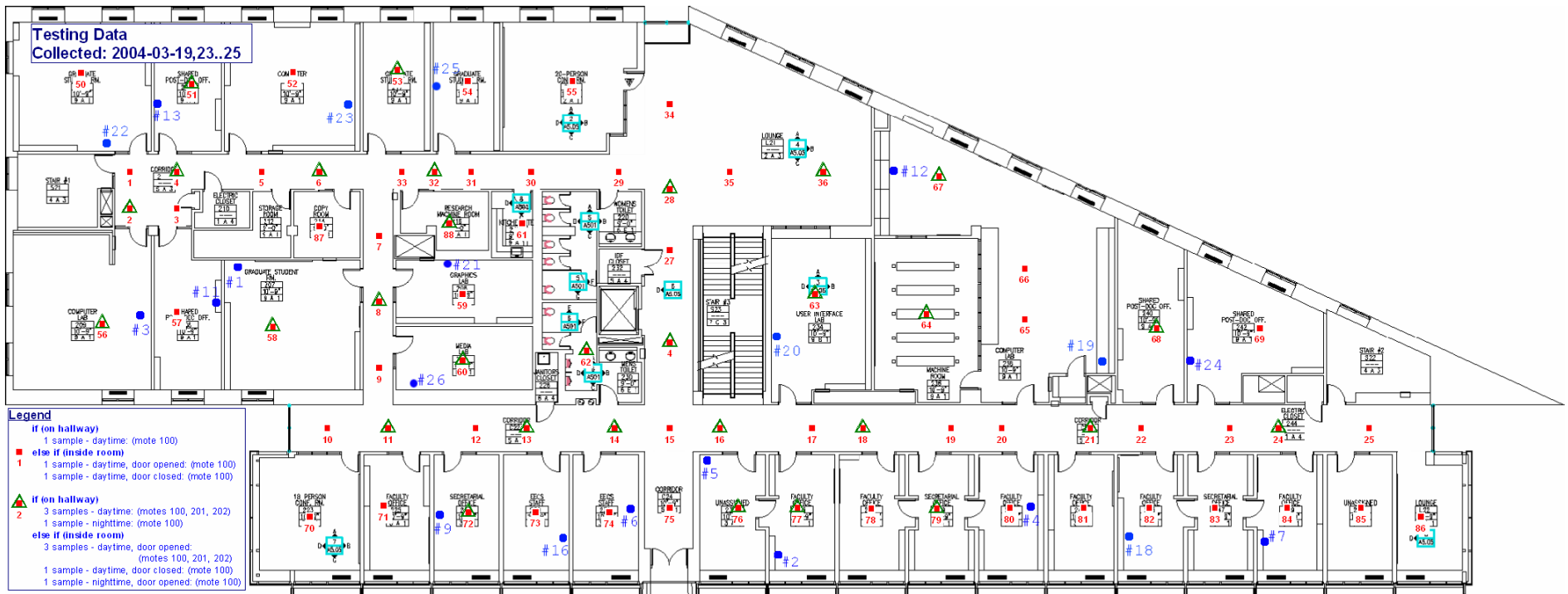
- Deployed on 2nd floor of MD (1,742 m²) using MoteLab
- 20 beacon nodes
- Collected 482 reference signatures throughout the floor over 5-days
 - Hallway, rooms, doors opened and closed, time of day, different mobile motes (at least 30 data points per experiment)
- Each signature was collected for 1 min. during which each beacon node transmitted at 4Hz cycling through 7 TX power levels (from -20 to 10 dBm)

Training Data

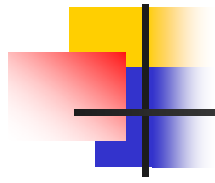


- Blue dots – beacon nodes
- Red squares – location of reference signatures
 - If inside room, then both door opened and closed

Testing Data



- Blue dots – beacon nodes
- Red squares – location of reference signatures
 - If inside room, then both door opened and closed
- Green triangles – multiple reference signatures collected
 - Different motes and time of day

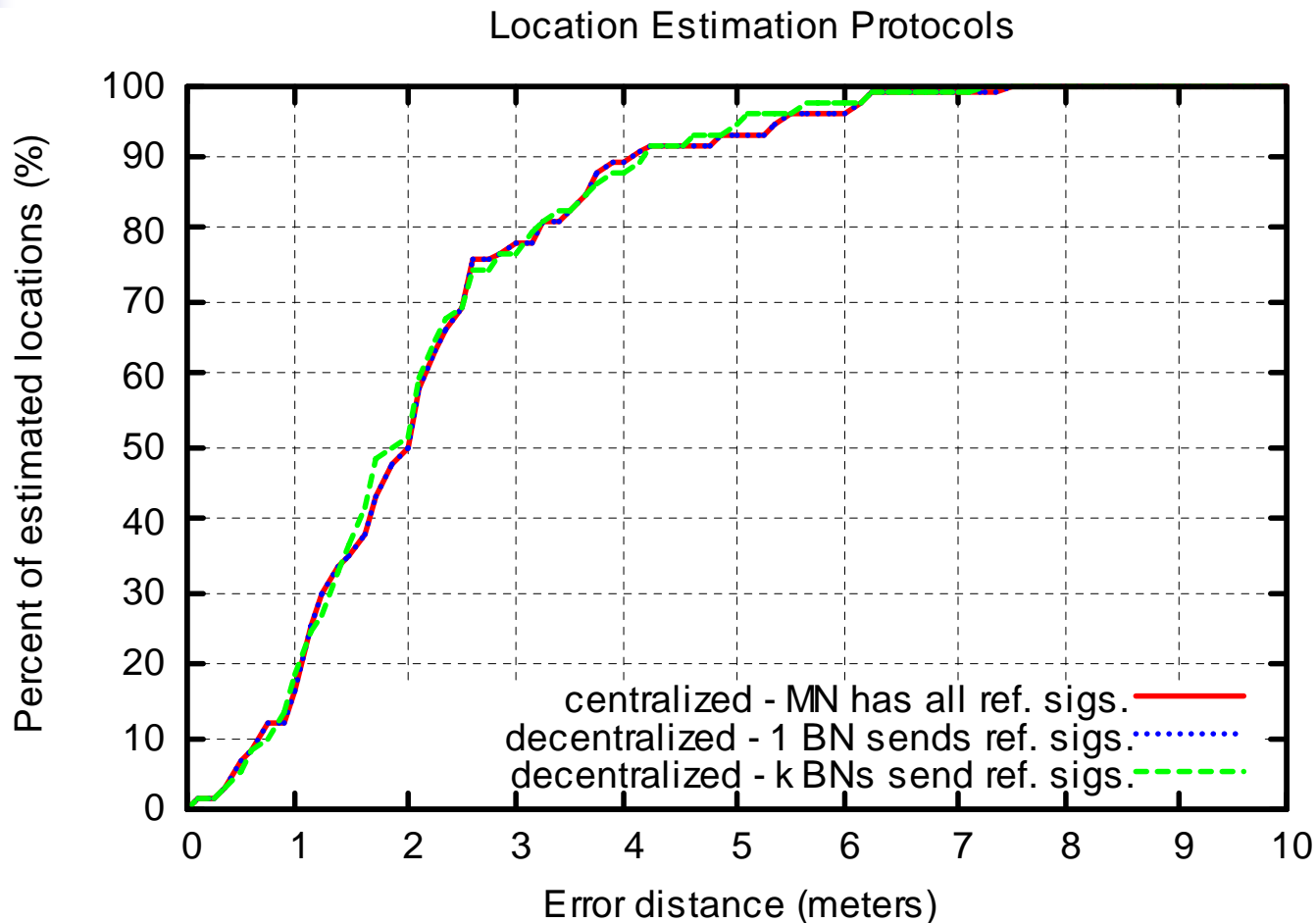


Evaluation

Evaluation Overview

- What are the effects of the various algorithms and parameters on accuracy?
 - Decentralized location estimation protocols
 - Selection of reference signatures
 - Number of TX powers
 - Density of reference signatures and beacon nodes
 - Distribution of the reference signature db. (greedy vs. balanced)
- How robust is MoteTrack with no beacon node failures?
 - Perturbation of RF signals
 - Daytime vs. nighttime
 - Manufacturing differences between motes
 - Hallway vs. rooms with doors opened and closed
- How robust is MoteTrack with beacon node failures?
 - Adaptive signature distance metric

Decentralized Loc. Estimation Protocols

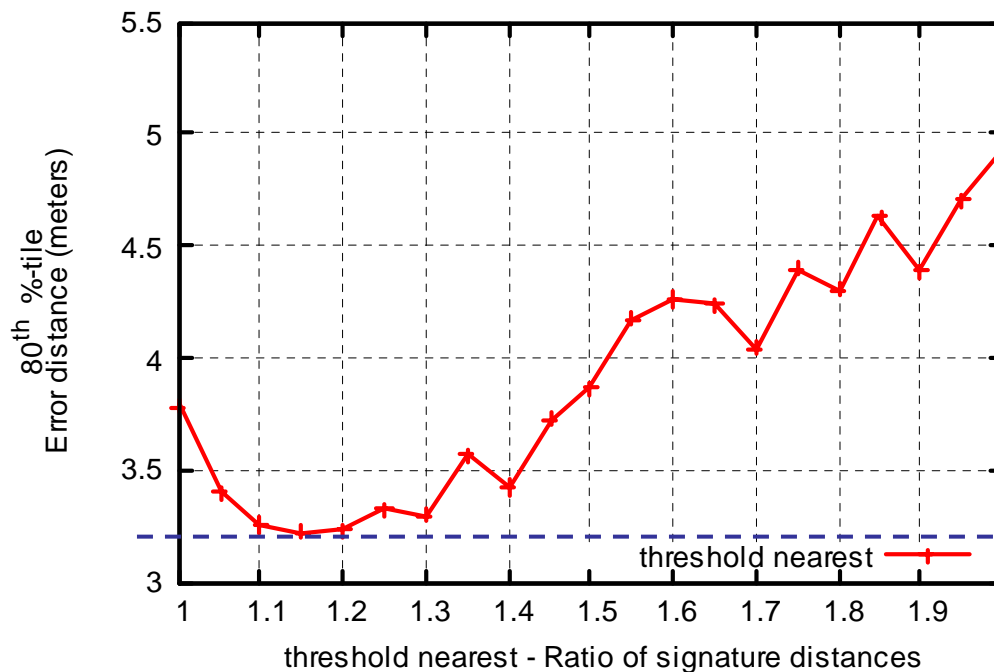


- Under normal circumstances, both decentralized algorithms perform nearly identical to the centralized version.

Selection of Reference Signatures

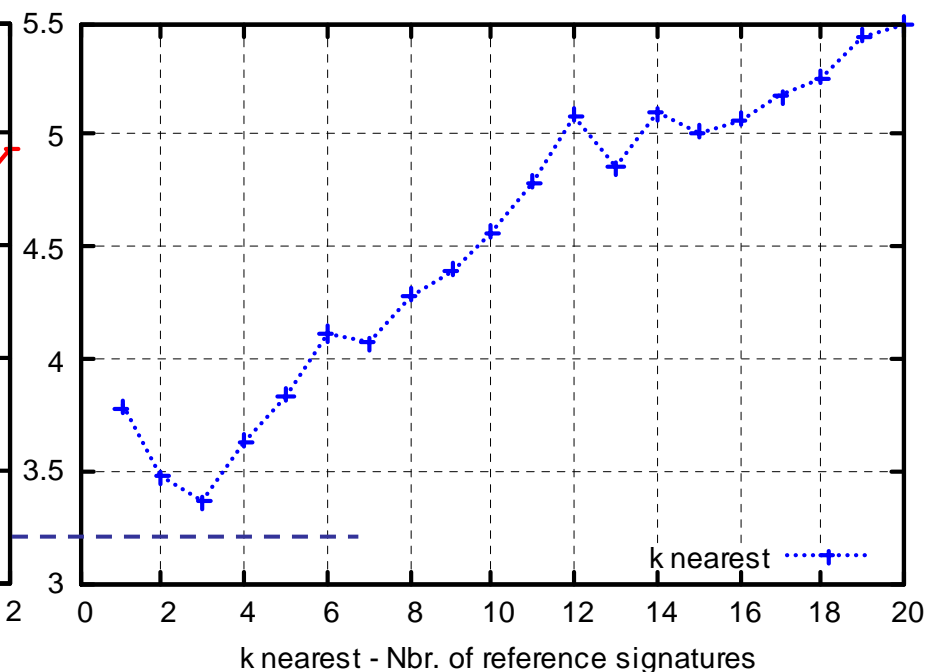
K nearest

Location Clustering Algorithm - k Nearest



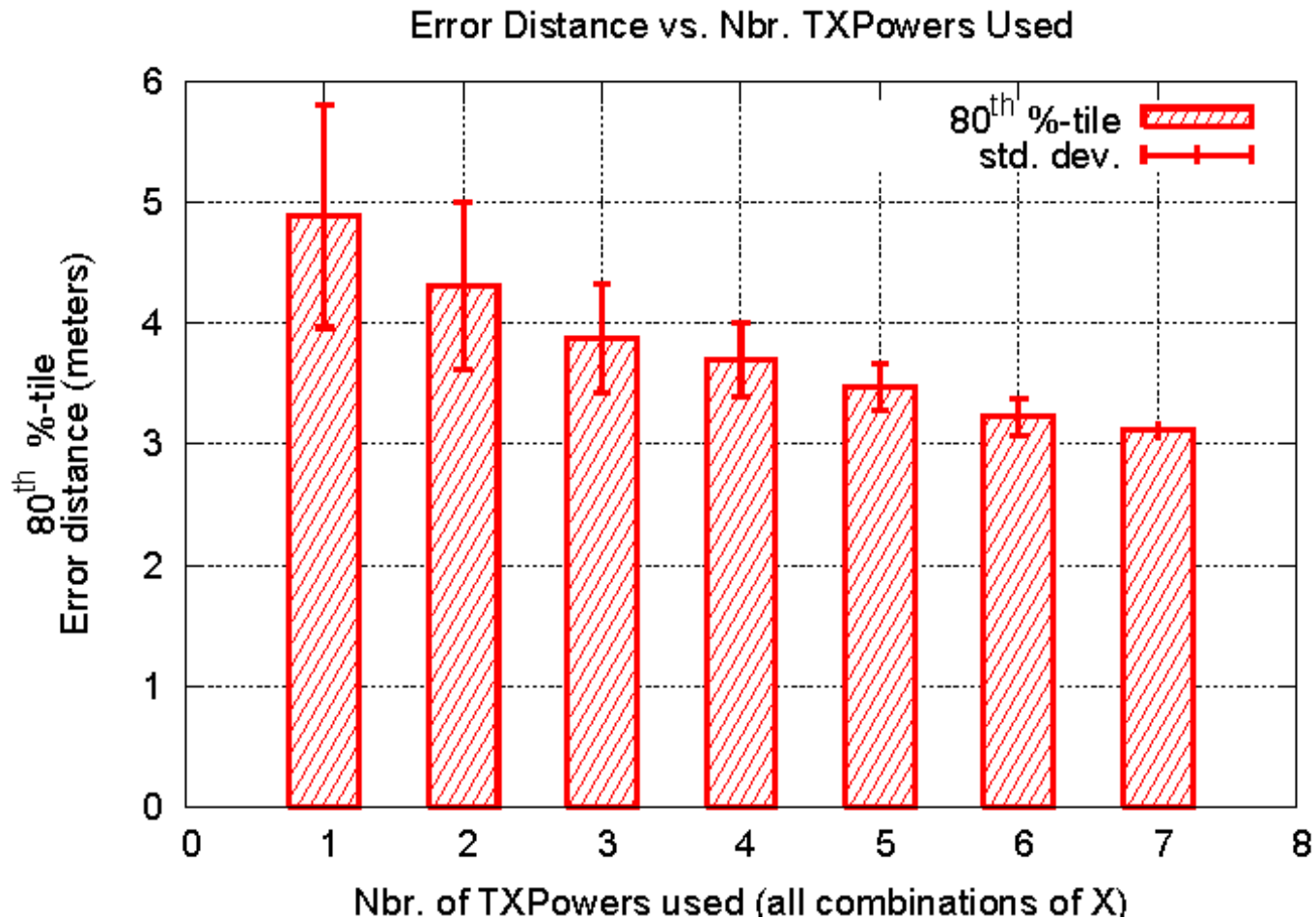
Threshold nearest

Location Clustering Algorithm - Threshold Nearest



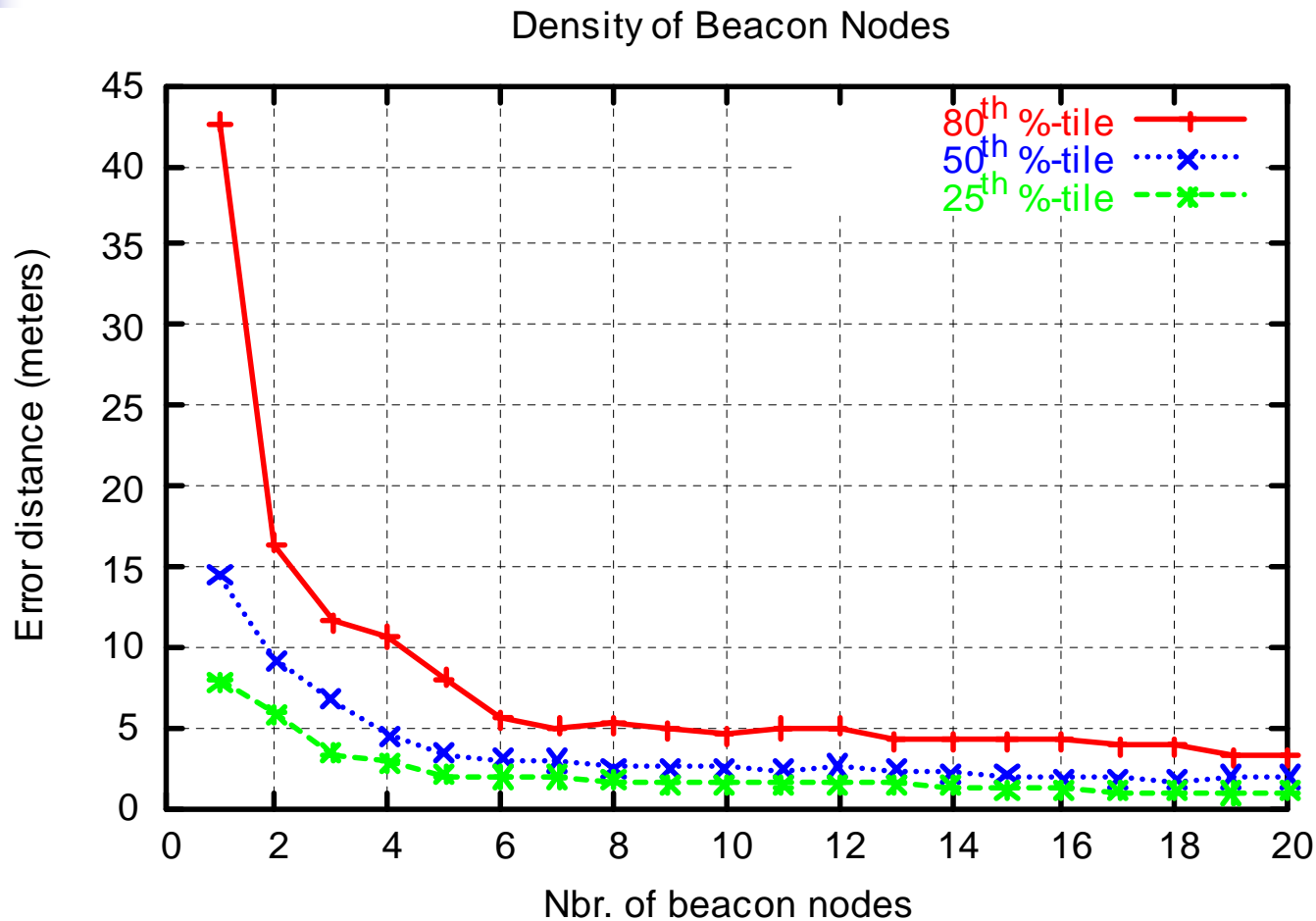
- The *relative threshold* algorithm outperform the *k nearest* one

Number of Transmission Powers



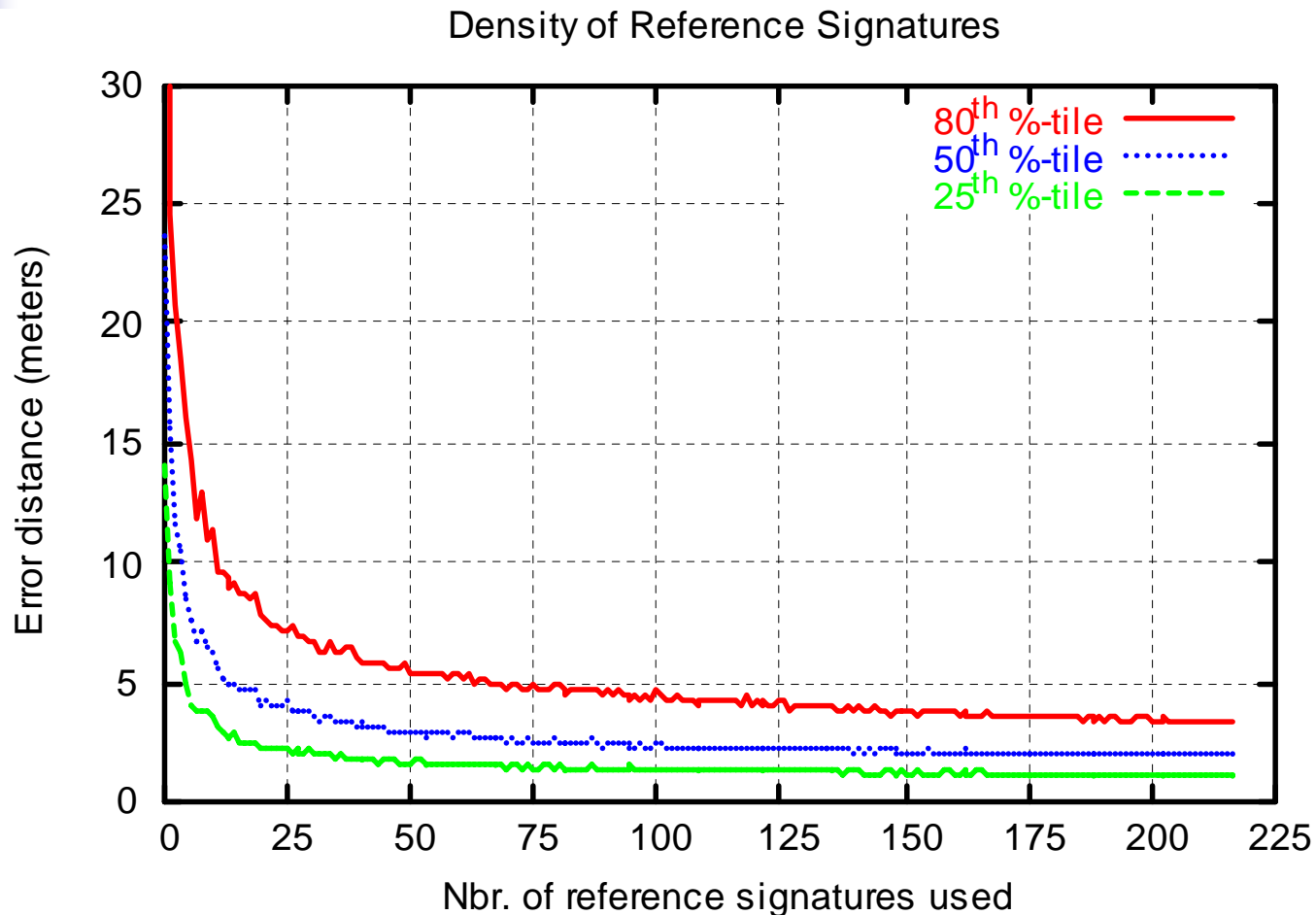
- Using more TX powers reduces error and std. deviation considerably
- TX powers range from -20 to 10 dBm

Density of Beacon Nodes



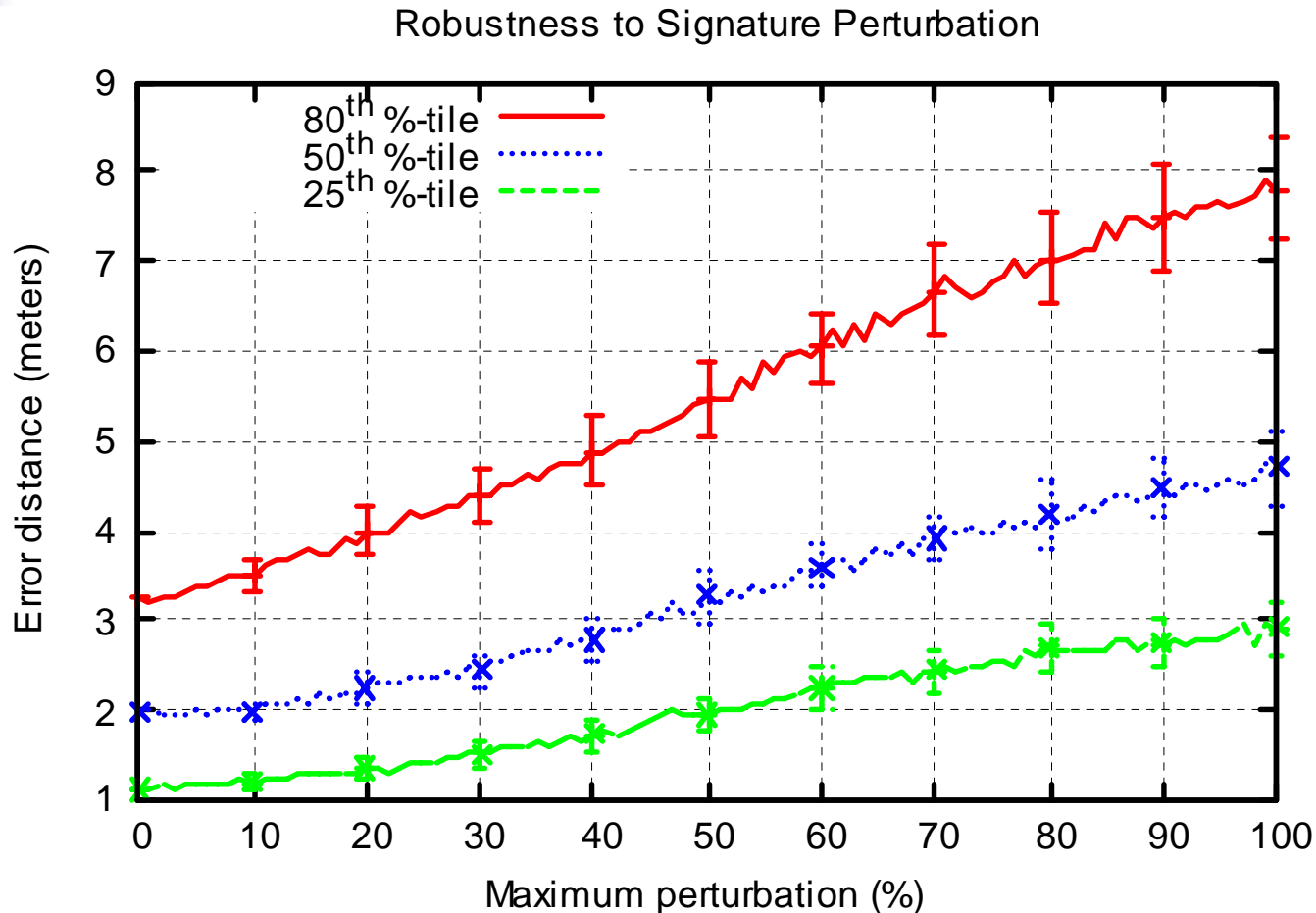
- Critical number is about 6-7 beacon nodes (0.004 beacon nodes/m²) (total area: 1,742 m² or 18,751 ft²)

Density of Reference Signatures



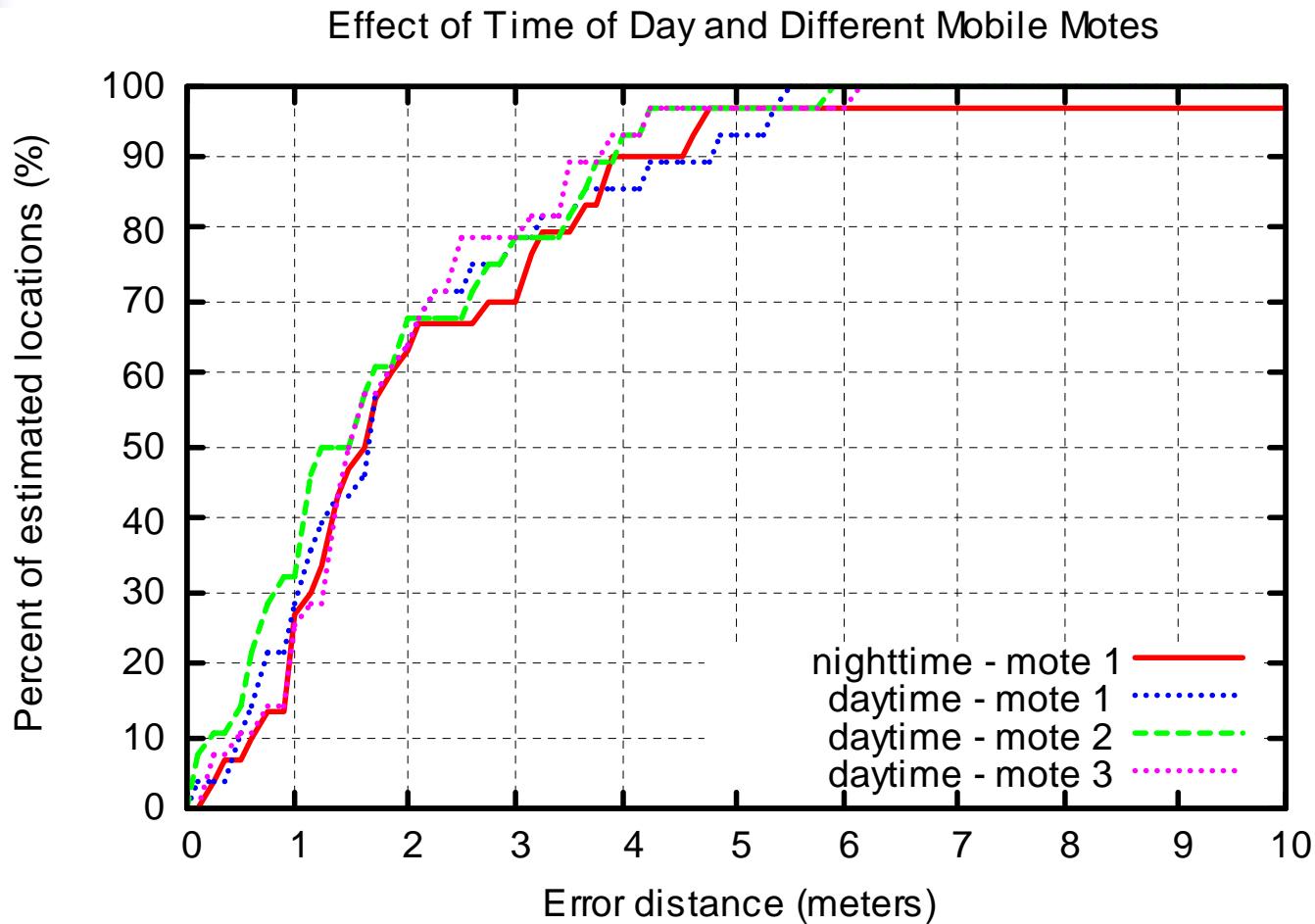
- After 75 reference signatures (0.043 ref. sigs./m²), the accuracy begins to stabilize with additional reference signatures (total area 1,742 m²)

Robustness to RF Perturbations



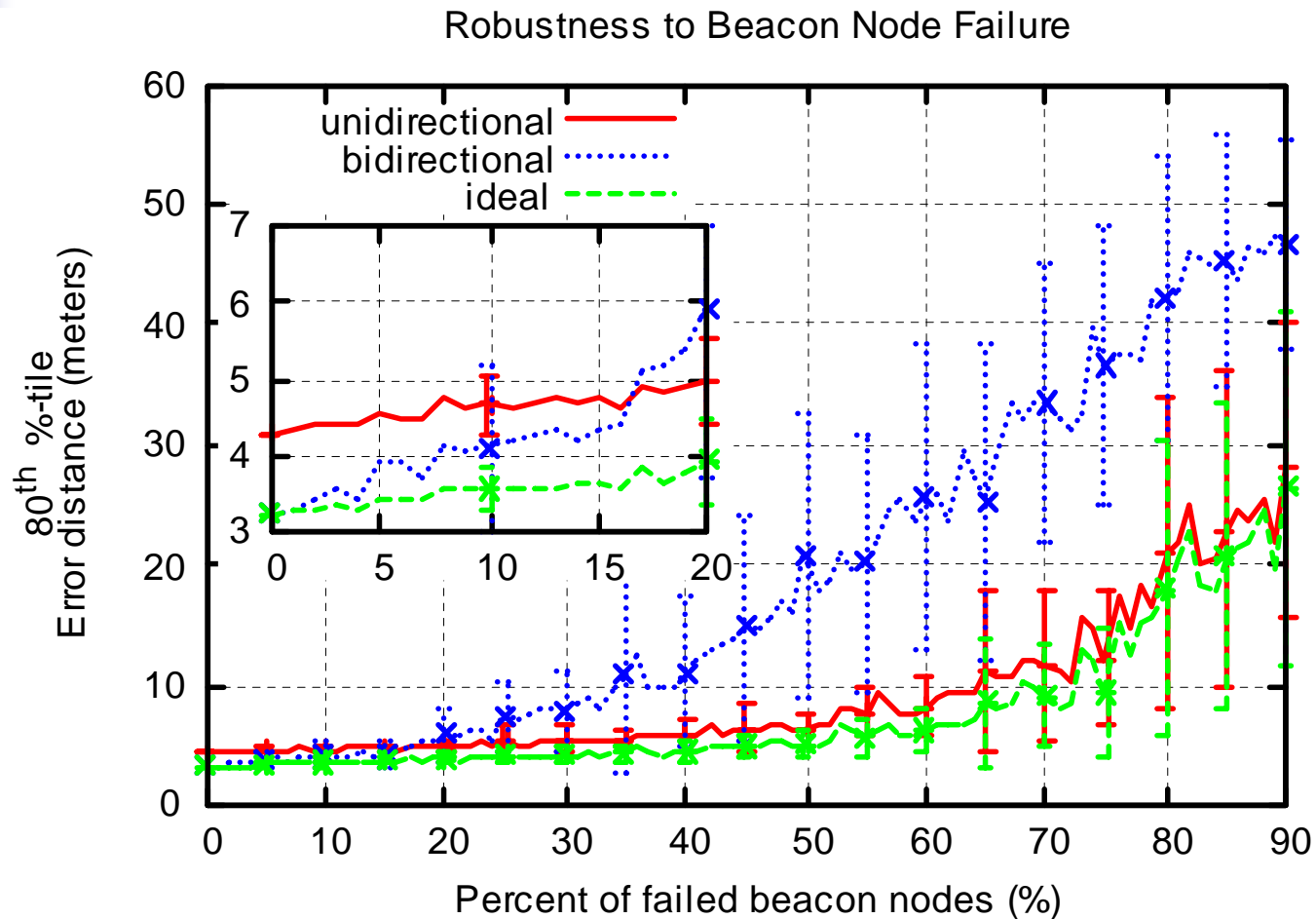
- Accuracy degrades linearly with increased perturbation levels
- For maximum perturbation of 40%, the 80th %-tile has an accuracy of under 5 meters

Time of Day and Different Motes



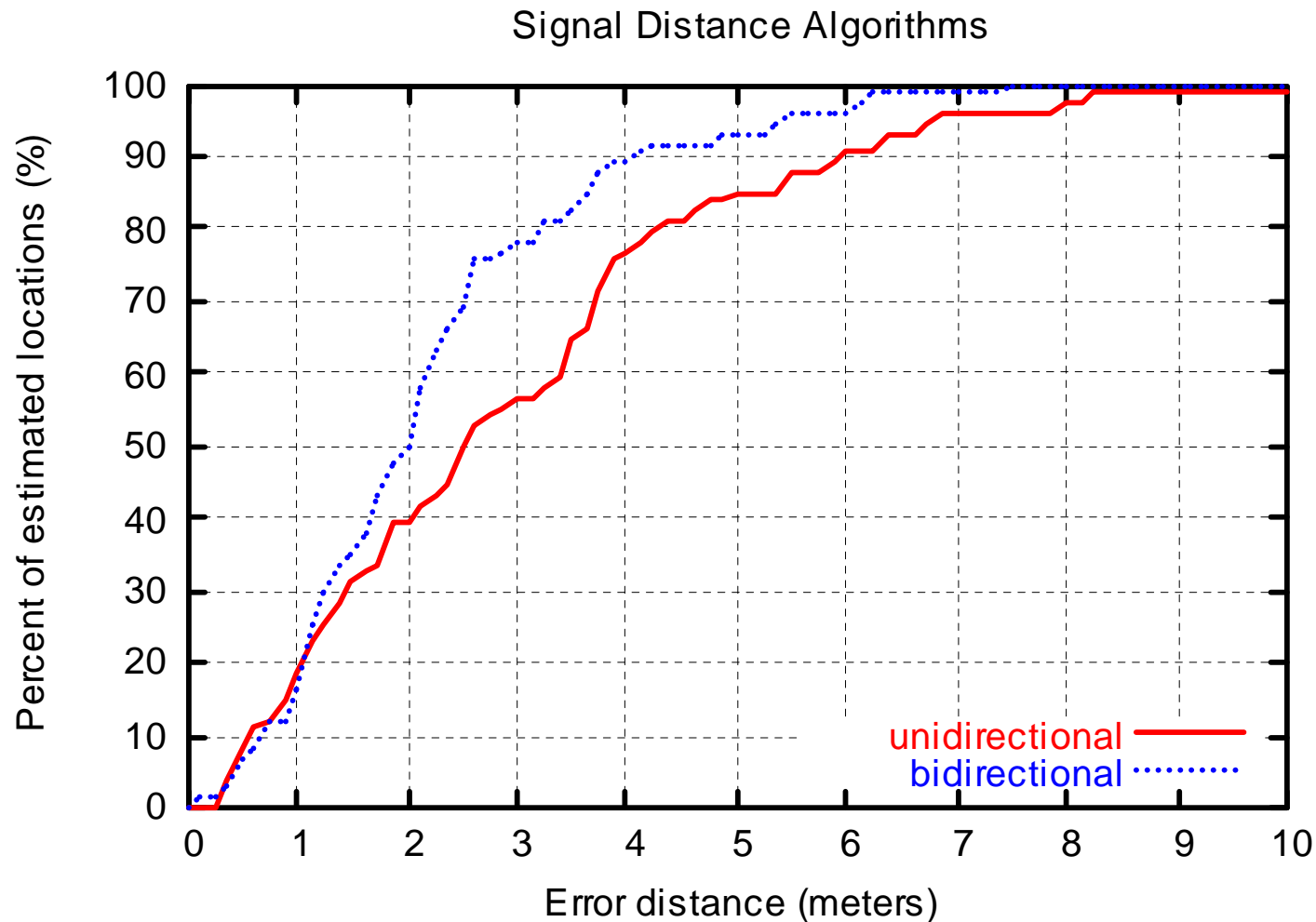
- No consistent differences are found when varying the time of day or the mobile mote

Robustness to Beacon Node Failure



- The unidirectional algorithm is more robust to significant beacon node failures, but yields poorer accuracy for few failures (less than 16%)
- MoteTrack uses a hybrid approach
- *ideal*: bidirectional metric only over non-failed nodes (req. perfect knowledge)

Signature Distance Algorithms

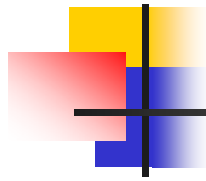


- The bidirectional algorithm is more accurate because the comparison space is larger



Outline

- Motivation
- Introduction
- System Description
- Implementation and Data Collection
- Evaluation
- **Future Work and Directions**
- **Conclusions**



Future Work and Directions

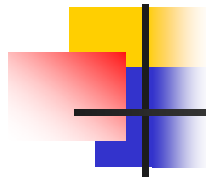
Ad hoc rapid deployment – automatic calibration via GPS

(e.g., on the scene of a train wreck)

- Add/remove reference signatures on-the-fly from beacon nodes
- How to build up the reference signature database?
 - Rescue personnel populate beacon nodes with reference signatures as they move about with GPS enabled PDAs
- Additional error introduced by the GPS

Immediate future

- Port to Telos and MicaZ motes which use the CC2420 802.15.4 radio chip
- Re-run measurements to compare performance of 433MHz to 2.4GHz



Conclusions

Main contributions

- Extended the basic RF signature approach to achieve high robustness by:
 - Decentralizing the location estimation protocol so that it relies only on local communication, data, and operational nodes
 - Replicating the ref. signature database across beacon nodes in a fashion that minimizes storage but achieves high level of robustness to failure
 - Using an dynamic signature distance metric that adapts to locally failed beacon nodes
- Negligible error introduced with up to 40% of beacon node failure
- Demonstrated a complete workable system and performed detail evaluation

Availability

- MoteTrack 1.0 has been released
 - Get the latest code from the TinyOS *contrib* directory on SourceForge
- MoteTrack web site:
 - <http://www.eecs.harvard.edu/~konrad/projects/motetrack/>