Analysis of Localization Algorithms for Sensor Networks

Jana van Greunen
Outline

- Introduction
- Algorithm overview
- Comparison & evaluation
- Extensions:
  - Real-time error estimation
  - Low-energy computation
  - Results
- Conclusion
Introduction

- Sensor network consists of many small, cheap, self-sustaining, densely deployed sensor nodes.
- Applications require that the nodes in the network are aware of their geographic location.
- Too expensive to use GPS on every node
- Thus need algorithms to compute each node’s position using node-to-node range measurements and information from a few reference nodes
- Range measurements are made between each node and its neighbors within a circular area
Introduction cont’

- Measurements are made using TOA or RSSI – both methods are error prone
- A good localization algorithm should:
  - Be tolerant to range errors
  - Scale with network
  - Minimize communication & computation energy spent
  - Converge rapidly and accurately
  - Perform well across network topologies
  - Provide a measure of error
Algorithm overview

- **Centralized LP (Doherty et al.)**
  - Key assumption: if two nodes can communicate with each other, they must lie within the communication radius $R$ of each other.
  - Mathematically = 2-norm constraint on the node positions

$$\|a-b\|_2 < R \Rightarrow \begin{bmatrix} I_2 R & a-b \\ (a-b)^T & R \end{bmatrix} \geq 0 \quad (xR)$$

- Combine local connectivity-induced constraints to solve:

$$\text{Minimize} : c^T x \\
\text{Subject to} : Ax < b$$
Rectangular intersection (Simič)

- Partition space into cubes/squares (cells)
- Communication area is a square (#cells)

At every unknown node:

**Step A:** Gather positions of one-hop neighbors with known positions

**Step B:** Compute estimated position via minimum rectangular intersection
Network Coordinate (Capkun)

2 phase algorithm

- Local coordinates
  - Each node $j$ measures distances to its one-hop neighbors and their distances from each other
  - Place one-hop neighbors in local coordinate system (trigonometry)

- Global coordinates
  - Align local coordinates
DV-Hop (Niculescu et al.)

- Use ‘average’ distance to prevent error propagation
- Known nodes flood ‘hops’ and position through network
- Each unknown node stores the position and hop-distance (# hops*average distance) from all the known nodes
- When an unknown node has its hop-distance from more than three non-collinear known nodes it can compute its position via triangulation (solve Ax=b)
- This algorithm works well when topology is regular
2-phase
  - Initial position estimate
    - DV-Hop
  - Refinement
    - Nodes try to improve their position estimates by iteratively measuring the distances to one-hop neighbors and then performing weighted maximum likelihood triangulation.
    - All unknown nodes start with a weight of 0.1
    - Known nodes have weight 1.0
    - After each position update the weight of the node is set to the average weight of all its neighbors
## Comparison

<table>
<thead>
<tr>
<th></th>
<th>Centralized LP</th>
<th>Rectangular intersection</th>
<th>Network Coordinate</th>
<th>DV-Hop</th>
<th>Start-up &amp; refinement</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Scalable</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Energy efficient</strong></td>
<td>No</td>
<td>Yes</td>
<td>Moderately</td>
<td>Yes</td>
<td>Moderately</td>
</tr>
<tr>
<td><strong>Accuracy</strong></td>
<td>Good</td>
<td>Spotty</td>
<td>Poor</td>
<td>Medium</td>
<td>Fair</td>
</tr>
<tr>
<td><strong>Speed of convergence</strong></td>
<td>Depends on network size</td>
<td>Fast</td>
<td>Depends on network size</td>
<td>Medium</td>
<td>Medium</td>
</tr>
<tr>
<td><strong>Tolerant to range error</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Moderately</td>
</tr>
</tbody>
</table>

Note: no algorithm provides a measure of final position error
Extensions

- **Real time error estimate for centralized algorithm (Patwari et al.)**
  - The Cramer-Rao Bound: lower bound on the covariance matrix of any unbiased estimator $\theta$.
  - Mathematical formulation:
    \[
    \text{Var}(\hat{\theta}) \geq [F^{-1}(\theta)]_{ii}
    \]
    \[
    [F(\theta)]_{ij} = -E\left[\frac{\partial^2 \ln p(x;\theta)}{\partial \theta_i \partial \theta_j}\right]
    \]
  - **Assumptions:**
    - $d_{ij}$’s are Gaussian distributed & independent for all $i,j$.
    - The variance $\sigma_d$ associated with $d_{ij}$ is independent of $|d_{ij}|$ and is the same for all measurements in the network.
Cramer-Rao bound cont’

- Using these assumptions, $p(x;\theta)_i$ is the multiplied densities of all distance measurements (Gaussian with mean $d_{ij}$ and variance $\sigma_d$)

- **Extend this to the distributed case:**
  - Each time unknown node $i$ estimates its position it calculates its own Cramer-Rao bound
  - However, information from unknown neighbors are used – thus need to account for the uncertainty in their positions
  - Solution: for each measurement $d_{ij}$ increase $\sigma_d$ to $\sigma_d + \sigma_j$
Low-power computation

- Idea: Create a low-computation, distributed and accurate algorithm.
- Combine computation of rectangular intersection with accuracy of start-up & refinement
- Replace least-squares triangulation with rectangular intersection in the start-up and refinement
- Computation savings: multiplies $\Rightarrow$ comparisons
- Also can have simpler hardware, because there are no multiplications
Results

Simulation:

- OMNET++ (network simulation package)
- Randomly placed 400 nodes in 100*100 rectangular area.
- Varied communication range from 5-15

- Anchors & connectivity still => better accuracy
- On average low-power computation performs two times worse than triangulation
Results cont’

- For low connectivity – roughly the same error
- High connectivity much poorer performance
- More anchors still ⇒ better accuracy
- Connectivity does not make big difference anymore
- This is because weights are not used and correlated errors prevent convergence
Future work

- Look at non-homogenous networks, perhaps some nodes should/are more equipped to do more computation
- Incorporate a few long distance measurements to collapse error
Conclusion

- Need a scalable, energy efficient solution
- There are many different existing localization algorithms
- Performance of distributed algorithms are heavily dependent on underlying network topology
- Developed an estimate of error in the position
- Trade-off between energy and accuracy
- The low-power start-up and refinement performs 2-3 times worse than the normal algorithm