Controlling Error Propagation in Mobile-Infrastructure Based Localization

Ying Zhang and Juan Liu

Intelligent Systems Laboratory
Palo Alto Research Center

MELT'09
September 30, 2009
Overview

Motivation
- UbiSense Real-Time Localization Systems (RTLS)
- Mobile Infrastructure Localization

Base Solution
- Leapfrog
- Refinements

Error Control
- Error control strategies
- Error modeling and estimation

Experiments
- Simulation
- Real platform

Conclusions
- Lessons learned
- Applications
Localization Overview

Infrastructure-based Localization (UbiComp)

Many static anchor nodes
One mobile node with
One hop to static anchor nodes

Self Localization (Sensor Networks)

Many static nodes
No anchors or
Multiple hops to anchor nodes
Reversed Roles of Tags and Readers in Ubisense Localization

Tags:
- Small
- Light
- Cheap

Readers:
- Large
- Heavy
- Expensive
Our Approach: Mobile Infrastructure Localization

What:
- Use mobile nodes to localize static nodes
- Mobile platform can have one or more nodes
- Mobile platform can be active or passive

Why:
- Increase accuracy
- Reduce cost

How:
- Adding extra connections
- Adding extra sensing modality
- Using cheap static nodes (permanent or ad-hoc landmarks)
  With reusable expensive platform
Innovation

*Mobile Localization Infrastructure for AOA,*

*Structured Multi-Reader Mobile Platform,*

*Constrained Motion*
Base Solution
\((x_t - x_s) \cdot \sin(a + \alpha) - (y_t - y_s) \cdot \cos(a + \alpha) = 0\)

\((x_t - x_s) \cdot \sin(\beta) - (z_t - z_s) \cdot \cos(\beta) \cdot \cos(a + \alpha) = 0\)
Leapfrog Algorithm

1. Compute the tag location from known cart locations (the first location is known), and mark the tag as known afterwards.
2. Compute the cart location from known tag locations and mark the cart location as known afterwards.
Compute Tag Locations from Cart Locations

Line intersection: a tag needs to be seen by at least 2 readers

\[
\begin{align*}
(x_t, y_t, z_t) &= \left[\begin{array}{c}
x_s \\
y_s
\end{array}\right] + \frac{1}{2} \left[\begin{array}{cc}
\cos(a + \alpha_1) & \cos(a + \alpha_2) \\
\sin(a + \alpha_1) & \cos(a + \alpha_2)
\end{array}\right] \left[\begin{array}{c}
\lambda_1 \\
\lambda_2
\end{array}\right] \\
\lambda_1 &= \lambda_2 \\
\lambda_2 &= \lambda_1 \\
\beta &= \beta_1 = \beta_2 \\
\alpha &= \alpha_1 = \alpha_2 \\
\tan(\beta) &= \frac{\lambda_1 \tan(\beta_1) + \lambda_2 \tan(\beta_2)}{2}
\end{align*}
\]
Compute Cart Location from Tag Locations (Linearization)

If a tag is seen by both readers in the cart, with minimum 2 such tags, i, j and reader k=1 or 2

\[
\lambda_{ik} \cos(\alpha_{ik}) - \lambda_{jk} \cos(\alpha_{jk}) = (x_i - x_j) \cos(a) + (y_i - y_j) \sin(a) \quad (1)
\]

\[
\lambda_{ik} \sin(\alpha_{ik}) - \lambda_{jk} \sin(\alpha_{jk}) = (y_i - y_j) \cos(a) - (x_i - x_j) \sin(a) \quad (2)
\]

Using Eq. (1) and (2) obtain \(\cos(a)\) and \(\sin(a)\)

\[
\begin{pmatrix}
\cos(a) \\
\sin(a)
\end{pmatrix} \leftarrow (A^T A)^{-1} A^T b
\]

\[
A = \begin{bmatrix}
x_2 - x_1 & y_2 - y_1 \\
y_2 - y_1 & -(x_2 - x_1) \\
\vdots & \vdots \\
y_n - y_{n-1} & -(x_n - x_{n-1})
\end{bmatrix}
\]

\[
b = \begin{bmatrix}
\cos(\alpha_{i_1}) \lambda_{2_1} - \cos(\alpha_{i_1}) \lambda_{1_1} \\
\sin(\alpha_{i_1}) \lambda_{2_1} - \sin(\alpha_{i_1}) \lambda_{1_1} \\
\vdots & \vdots \\
\sin(\alpha_{i_n}) \lambda_{2_1} - \sin(\alpha_{(n-1)_i}) \lambda_{(n-1)_1}
\end{bmatrix}
\]

\[a \leftarrow \arctan(\sin(a), \cos(a))\]

\[x \leftarrow \frac{1}{2n} \sum_{i=1}^{n} (x_i + \lambda_{i_1} \cos(a + \alpha_{i_1}) + x_i + \lambda_{i_2} \cos(a + \alpha_{i_2}))\]

\[y \leftarrow \frac{1}{2n} \sum_{i=1}^{n} (y_i + \lambda_{i_1} \sin(a + \alpha_{i_1}) + y_i + \lambda_{i_2} \sin(a + \alpha_{i_2}))\]

\[z \leftarrow \frac{1}{2n} \sum_{i=1}^{n} (z_i + \lambda_{i_1} \tan(\beta_{i_1}) + z_i + \lambda_{i_2} \tan(\beta_{i_2}))\]
Problems with Leapfrog

- Error Propagation
- Partial Solution

Approach: Leapfrog as Initial Solution to Refinement

\[
\min \sum (e_\alpha^2 + e_\beta^2)
\]

\[
(x_t - x_s) \cdot \sin(a + \alpha) - (y_t - y_s) \cdot \cos(a + \alpha) = e_\alpha
\]

\[
(x_t - x_s) \cdot \sin(\beta) - (z_t - z_s) \cdot \cos(\beta) \cdot \cos(a + \alpha) = e_\beta
\]
Error Control
### Error Control (A Complimentary Approach to Refinements)

Where do errors come from?

- Sensor reading error propagation
- Node error due to previous estimation

How to control error propagation?

- *Estimate* error for each of the location estimate, so that:
  - *Pick which sensor frame to localize first,*
  - *Instead* of selecting *all* known nodes as anchors, select only a *subset* of anchors with small error propagation property,
  - *Update* location estimate only if it has smaller error than previous estimate
Algorithm Flow Chart

Select first sensor frame

New known sensor frame?

yes

Select connected known sensor frames

For each connected tag

no

Update location and error

Select connected known tags

Estimate frame location

Compute frame location

Select next unknown frame

Update location and error

Done

Compute tag location

Estimate tag location error
Error Estimation

\[
\begin{bmatrix}
\lambda_1 \\
\lambda_2
\end{bmatrix} = \frac{d}{\sin(\alpha_2 - \alpha_1)} \begin{bmatrix}
\cos(\alpha_2) \\
\cos(\alpha_1)
\end{bmatrix} \Rightarrow \text{cov}(\lambda_1, \lambda_2) = \frac{d^2 \sigma^2}{\sin^2(\alpha_2 - \alpha_1)} \begin{bmatrix}
\sin^2(\alpha_2) & 0 \\
0 & \sin^2(\alpha_1)
\end{bmatrix}
\]

\[
B = \frac{1}{2} \begin{bmatrix}
\cos(a + \alpha_1) & \cos(a + \alpha_2) \\
\sin(a + \alpha_1) & \sin(a + \alpha_2)
\end{bmatrix}
\]

\[
\Omega^\lambda = B \cdot \text{cov}(\lambda_1, \lambda_2) \cdot B'
\]

\[
\Omega^{\alpha_1} = \frac{\lambda_1^2 \sigma^2}{4} \begin{bmatrix}
\sin^2(a + \alpha_1) & \cos^2(a + \alpha_1)
\end{bmatrix}
\]

\[
\Omega^{\alpha_2} = \frac{\lambda_2^2 \sigma^2}{4} \begin{bmatrix}
\sin^2(a + \alpha_2) & \cos^2(a + \alpha_2)
\end{bmatrix}
\]

\[
\Omega = \Omega^v + \Omega^\lambda + \Omega^{\alpha_1} + \Omega^{\alpha_2}
\]

\[
e = \text{trace}(\Omega)
\]
Experiments
Simulation Scenarios

hallway

ceiling

walls
Hallway Results

Leapfrog

Error Control

Refinement

parc
Palo Alto Research Center
Hallway Results (continued)

- **Log(s)**
  - Localization Time
  - Localization Accuracy
  - Error in 90% confidence

Graphs showing the fraction of error less than abscissa vs location error for different methods: Leapfrog, LeapErrCon, Leapfrog-RF, and LeapErrCon-RF.
Wall Results

Leapfrog

Refinement

Error Control

Localization Time

Localization Accuracy

parc
Palo Alto Research Center
Ceiling Results

Leapfrog

Refinement

Error Control

Localization Time

Localization Accuracy

<table>
<thead>
<tr>
<th>Error in 90% confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.6</td>
</tr>
</tbody>
</table>

parc
Palo Alto Research Center
Real Experiments
Conclusions
Challenges and Lessons Learned

Challenges

- For deployment over a large area, it requires tags distributed with enough density to have multiple tags be seen by the sensor frame at a time,
- Error control reduces the error propagation for large networks, only if the error is correctly modeled, which is hard for a real platform.

Remedies

- Using robotic platform with odometry
- Using additional sensors, such as cameras, LIDAR, to obtain extra data for localization
- Using floor-maps and anchor tags (e.g. with known heights)
Applications
Mobile Entity Localization and Tracking

Asset Tracking
Shipping
Warehouses
Robotics
Manufacturing
Questions?

Thank You