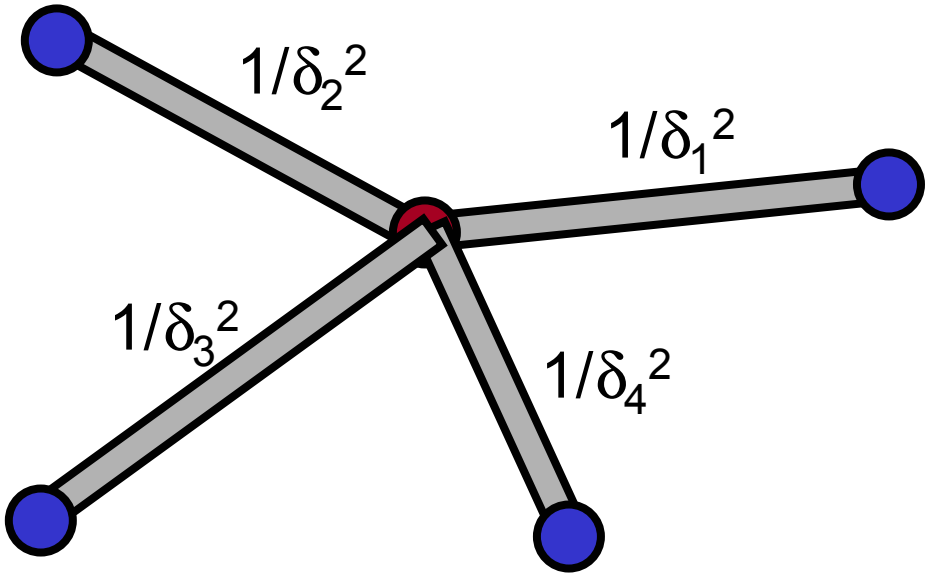
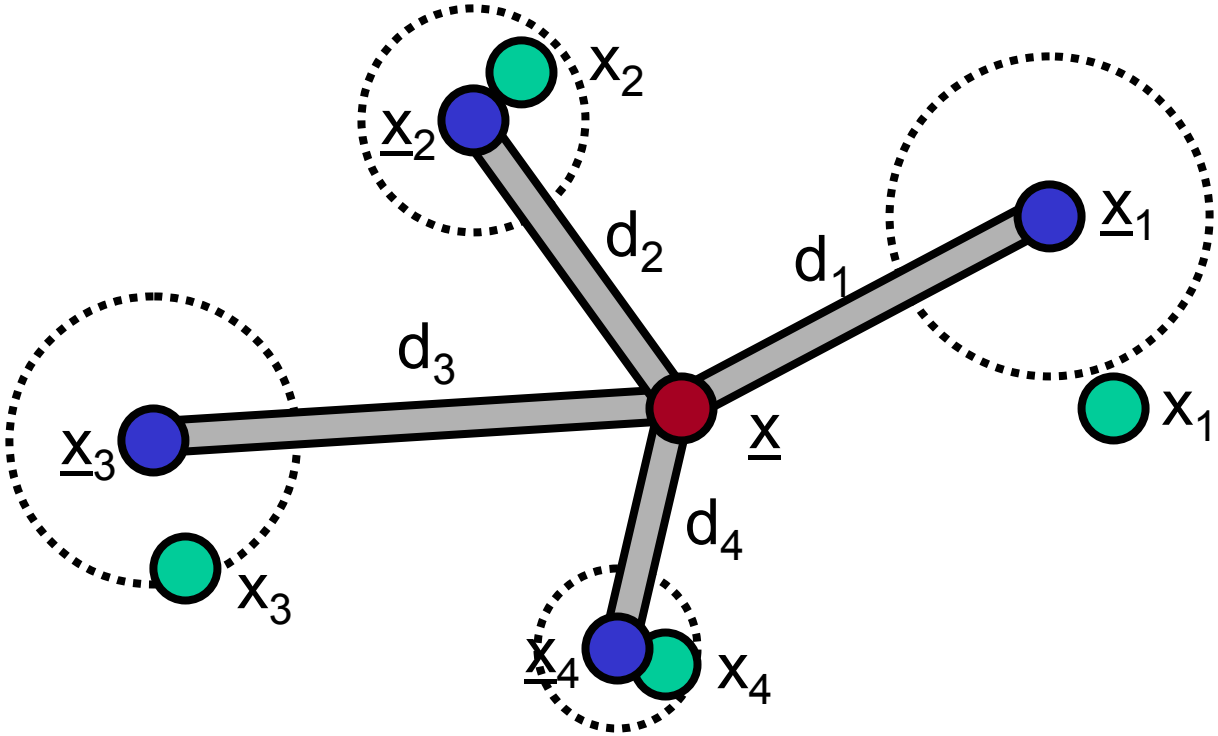
A Simple Sensor Node Localization Algorithm



Jeffrey Wu
CS428 Project
June 5, 2003

Consider the following localization problem:



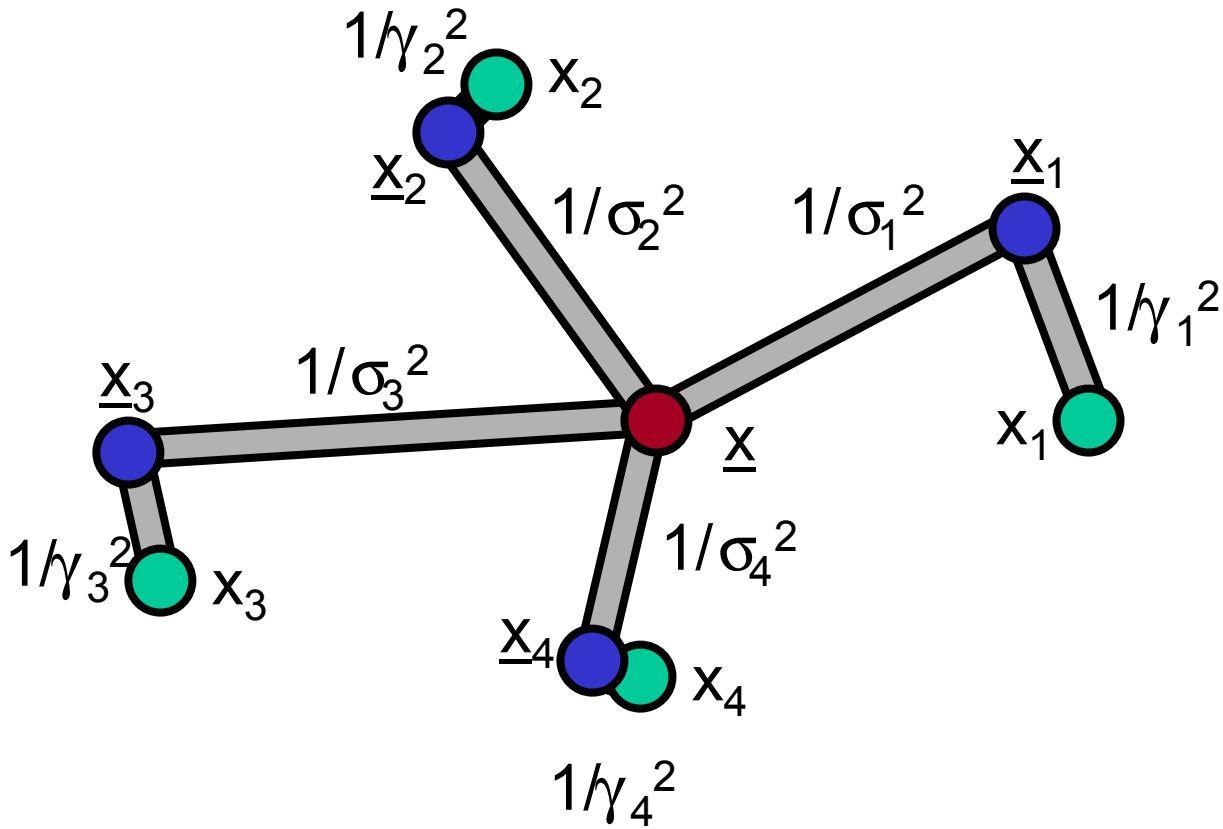
- $\underline{x}$ – Our actual position.
- $\underline{x}_i$ – Actual position of node i.
- x_i – Given position of node i, ~ $N(\underline{x}_i, \gamma_i^2 I)$.
- d_i – Measured distance to node i, ~ $N(\underline{d}_i, \sigma_i^2)$, $\underline{d}_i = ||\underline{x} - \underline{x}_i||$.

Problem : Find ML estimate of our position, x_{ML}.

Negative log-likelihood function:

$$L(\bar{x}, \bar{x}_1, \ldots, \bar{x}_n) = \sum_i \frac{\|x_i - \bar{x}_i\|^2}{2\gamma_i^2} + \frac{(\|\bar{x}_i - \bar{x}\| - d_i)^2}{2\sigma_i^2}$$

Physics analogy: $0.5K(d - \underline{d})^2$ is the potential energy of a spring with constant K, length d, and natural length $\underline{d}$.



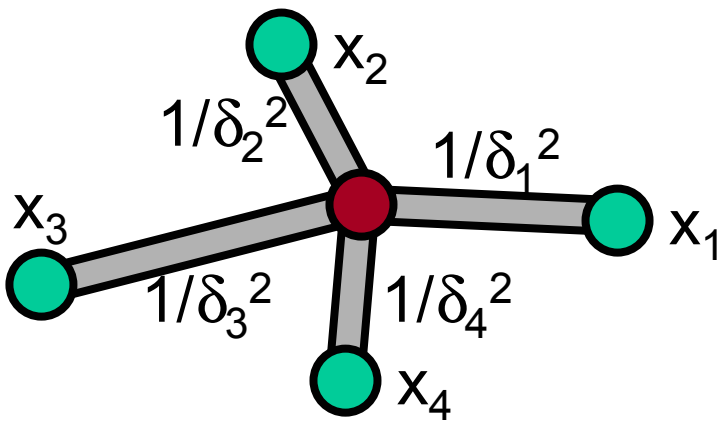
ML estimate of $\underline{x}, \underline{x}_1, \ldots, \underline{x}_n$ is ***minimum energy configuration***.

It is easy to see that in a minimum energy configuration, $\underline{x}$, $\underline{x}_i$, and x_i is ***colinear*** for each i. Using this fact, we can readily show that x_{ML} is the ML estimate iff it minimizes

$$L(x) = \min_{\bar{x}_1,\ldots,\bar{x}_n} L(x, \bar{x}_1, \ldots, \bar{x}_n) = \sum_i \frac{(\|x_i - x\| - d_i)^2}{2\delta_i^2}$$

where $\delta_i^2 = (\sigma_i^2 + \gamma_i^2)/2$.

Interpretation: We treat the given positions $x_1, \ldots, x_n$ as the actual positions and ***transfer the error*** in $x_1, \ldots, x_n$ to the d_i's



An approximate error analysis

Let $\underline{L}(x)$ be the (negative) log-likelihood when all our measurements are perfect, i.e. $x_i = \underline{x}_i$ and $d_i = \underline{d}_i$. Then $\underline{L}(x)$ achieves a minimum of 0 at $x = \underline{x}$. Thus the second-order approximation of $\underline{L}$ is

$$\underline{L}(x) \approx \tfrac{1}{2} (x - \underline{x})^T \nabla^2 \underline{L}(\underline{x})(x - \underline{x})$$

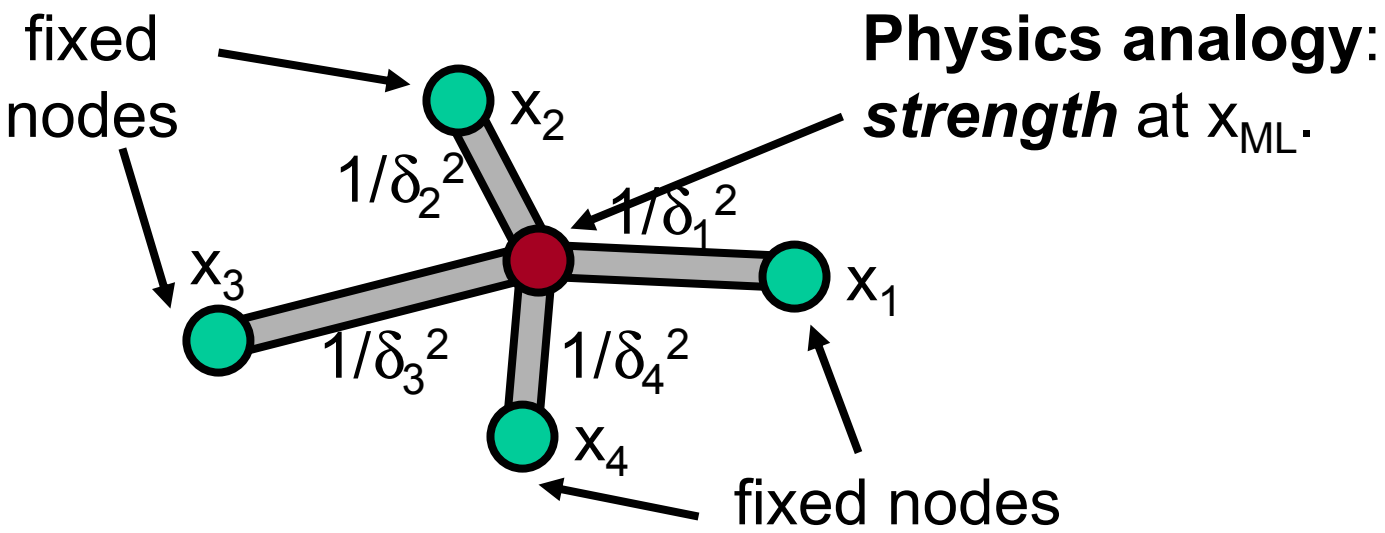
Thus the level set $\{x : \underline{L}(x) = c\}$ is approximately described by the ellipsoid $\{x : \tfrac{1}{2} (x - \underline{x})^T \nabla^2 \underline{L}(\underline{x})(x - \underline{x}) = c\}$, and so we can say that on this ellipsoid

$$\| x - \underline{x} \|^2 \leq 2 c / \lambda_1(\nabla^2 \underline{L}(\underline{x}))$$

Tightness of an optimal solution

Let x^* minimize the function f(x). We define the ***tightness*** of x^* as $\lambda_1(\nabla^2 f(x))$.

• The tightness measures the robustness of the solution to changes in f. Suppose that we wish to minimize f + g. Then the solution can deviate from x^* by at most about $\|\nabla g(x^*)\| / \lambda_1$.



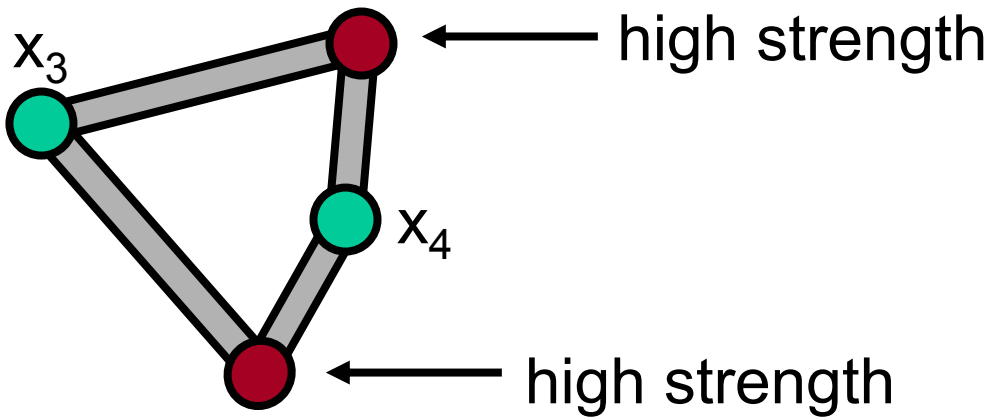
Solving the ML problem

Use Newton's method with backtracking line search.

- Extremely fast convergence.
- Cheap to compute because dimensions are small (2).

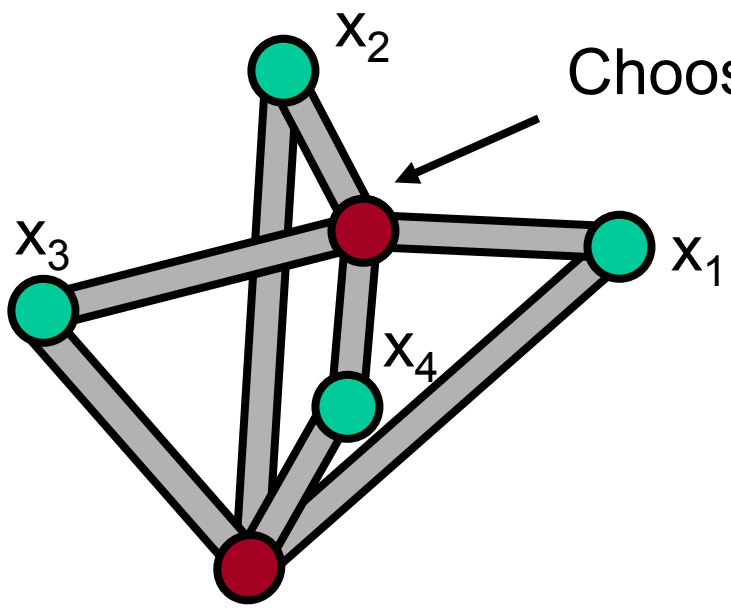
Choosing a good initial solution for Newton's method:

- Choose pair of nodes that maximizes **tightness** of solution with respect to only those nodes.



Yields ***two*** initial guesses. Need to eliminate one.

• Evaluate the value of L(x) at both proposed initial solutions and choose the one with least value, i.e. use ML detection.



Is possible to choose ***wrong*** initial estimate (i.e. one that falls into the wrong local minimum). Thus

- We require that difference in log-likelihood be large (say at least 10) before making decision.

A Sensor Node Localization Algorithm

- Each sensor i that has an estimate x_i also has a error estimate γ_i.
- Each sensor has a distance estimate $d_{ij} \sim N(\underline{d}_{ij}, \sigma_{ij}^2)$.
- A sensor i computes its ML estimate x_{ML} with respect to its neighbors, assuming that $x_i \sim N(\underline{x}_j, \gamma_j^2 I)$. If x_{ML} is ***unique*** we then broadcast

$$x_i = x_{ML}$$
$$\gamma_i^2 = L(x_{ML}) / \lambda_1 + 1 / \lambda_1$$

Mostly empirical formula, but …

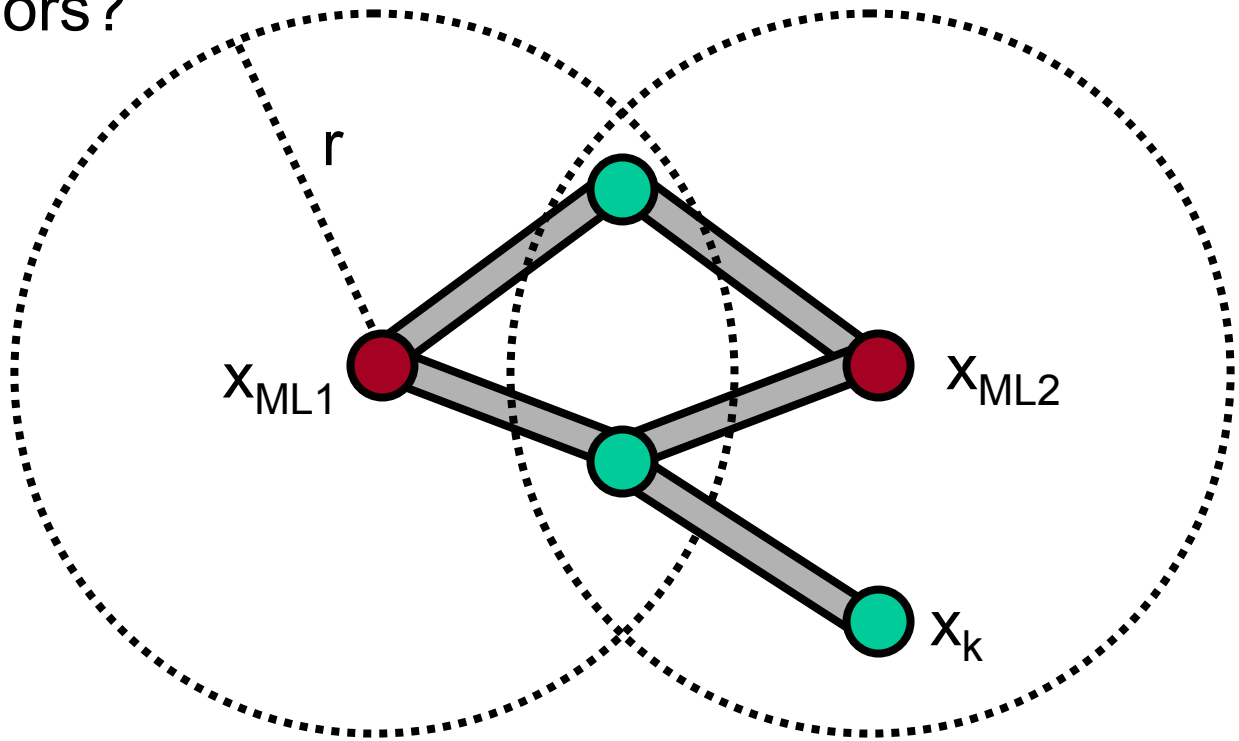
Inspired by 5th slide

Inspired by Cramer-Rao

Preventing Catastrophic Errors: We refuse to broadcast if γ_i exceeds a certain fixed threshold.

Increasing Node Participation

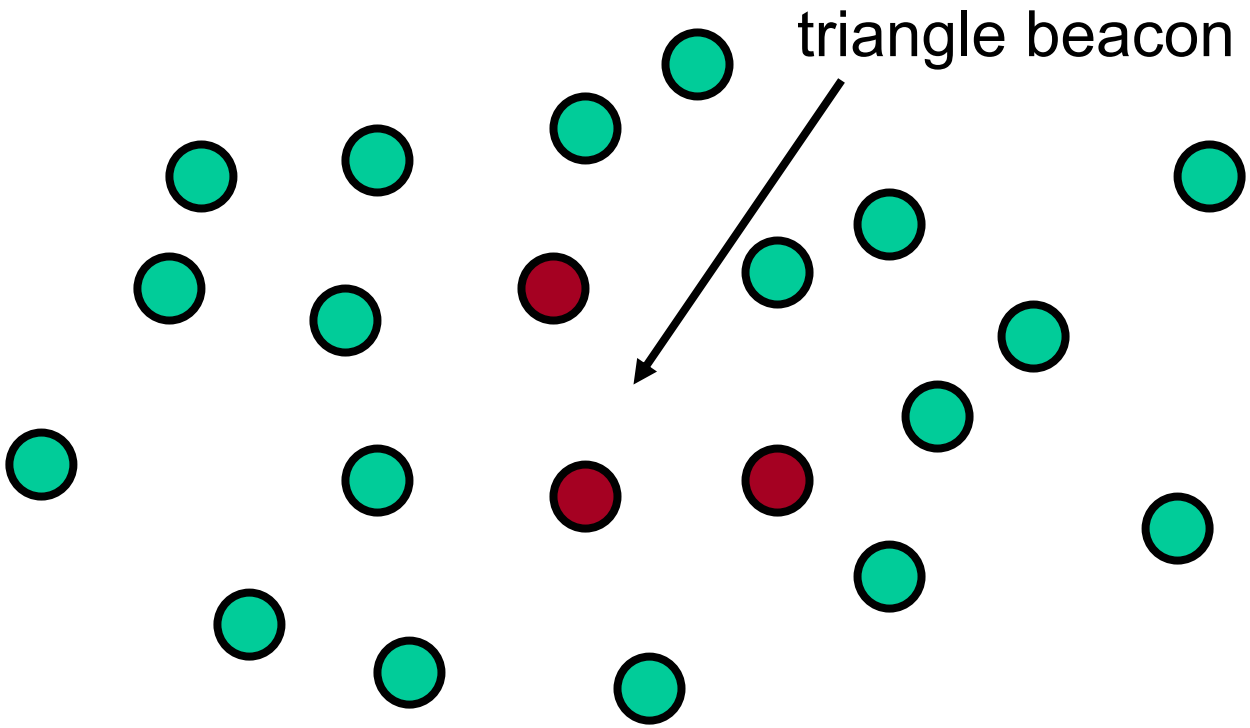
A node generally needs 3 neighbors with estimates to have a unique ML estimate. But what if it only has 2 neighbors?



The presence of node x_k is a ***certificate*** that x_{ML2} is not a possible location for x.

Triangle Beacons

Each node still needs three nodes within fairly close proximity to get a unique estimate (this time needing only two of them needing to be neighbors). Thus we deploy beacons in groups of three.



Simulation Results

Avg. distance between nodes = 15 m.
Distance measurement std. dev. = 0.02 m.
4 groups of 3 beacons, randomly placed in each simulation.

200 nodes :

Sensing range	Num. neighbors	Num. Estimates	Avg. error	Std. Dev.
18	5.3	93	0.055	0.066
19	5.78	27	0.0064	0.015
20	6.6	120	0.027	0.028
21	7.43	178	0.051	0.052
22	7.86	176	0.041	0.051
23	9.01	182	0.020	0.020
24	8.81	200	0.045	0.051
25	9.66	195	0.046	0.066
26	10.49	199	0.025	0.030

400 nodes:

Sensing range	Num. neighbors	Num. Estimates	Avg. error	Std. Dev.
18	5.7	70	0.0066	0.0088
19	6.26	129	0.043	0.053
20	6.955	66	0.025	0.034
21	7.55	270	0.126	0.131
22	7.89	294	0.056	0.060
23	8.705	375	0.077	0.068
24	9.52	203	0.048	0.043
25	9.94	400	0.062	0.050
26	11.21	3.86	0.171	0.165