Wireless Localization Using Self-Organizing Maps

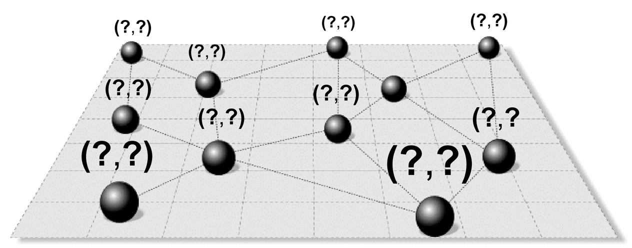
Dipartimento di Elettronica e Telecomunicazioni
Università degli Studi di Firenze, Firenze, ITALY

Gianni Giorgetti, Sandeep K.S. Gupta, Gianfranco Manes

IMPACT Lab, School of Computing and Information
Arizona State University, Tempe, US

1. Introduction

Knowledge of node positions is required in most WSN applications to interpret results. Sensor data are meaningless unless stamped with the location of the sensor that collected it.

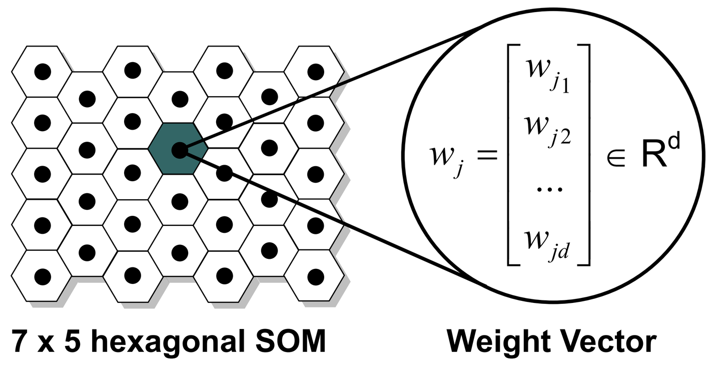


The neural network formalism of **Self-Organizing Maps** is used to implement a localization service.

- The scheme works using connectivity information only. Since it does not require range measurements (**range-free**) or anchor nodes (**anchor-free**), it is suitable for resource-constrained networks.
- The solution proposed achieve good localization results in **networks with low connectivity**, which are harder to localize, and in presence of irregular radio pattern or anisotropic deployment.
- The scheme has **low computation and communication overheads**. The algorithm can be executed on nodes with limited hardware resources .

2. Self-Organizing Maps

A Self-Organizing Map (**SOM**) is a neural network with the neurons arranged in a regular 2D lattice.



$$w_j = \begin{bmatrix} w_{j1} \\ w_{j2} \\ \dots \\ w_{jd} \end{bmatrix} \in R^d$$

Each neuron contains a weight vector. The weights, initialized with random values, will eventually store the information learned by the map.

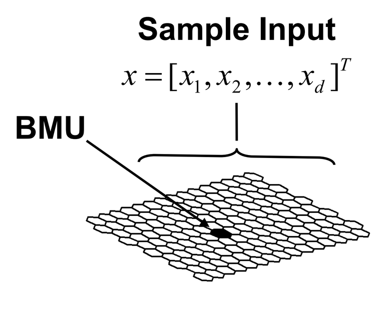
The map is trained using multiple iterations of a three step algorithm:

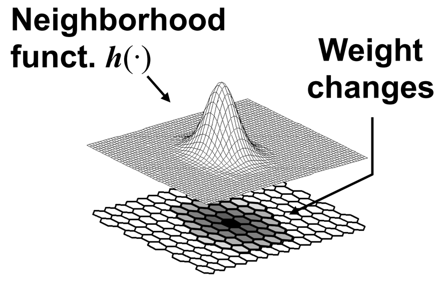
- **Sampling**: a sample is extracted from the input set and compared with the map weights.
- **Competition**: the neuron whose weights are more similar to the input is the Best Matching Unit (BMU).

$$BMU = \arg\min_j \|x - w_j\|$$

- **Adaptation**: the SOM weights are updated. The level of adaptation is regulated by a Gaussian shaped neighborhood function centered on the BMU.

$$w_j(n+1) = w_j(n) + \eta(n) h_{j,bmu}(n)[x - w_j(n)]$$





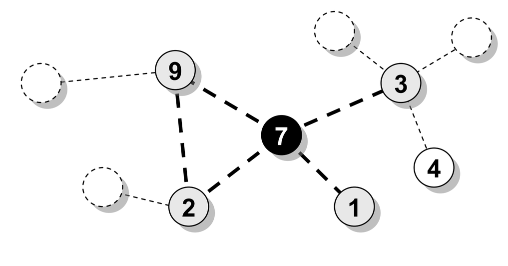
At each iteration, the learning parameter $\eta(n)$ and the width of the neighborhood function are decreased monotonically, allowing the map weights to converge to a stable configuration.

3. Topological Properties

SOMs are able to learn the underlying features of the input space. Final weights are **topologically ordered** (adjacent neurons converge to similar values).

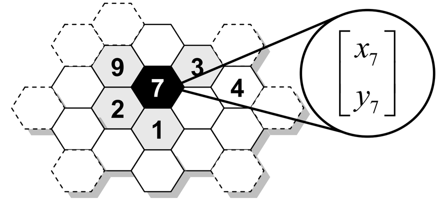
Example: a 10 x 10 SOM trained with random samples from the 3D RGB color space. After training, similar colors are mapped on nearby neurons.

SOMs can be used to solve the localization problem, which requires that radio neighbors are assigned to coordinates close in space.



Nodes in a WSN

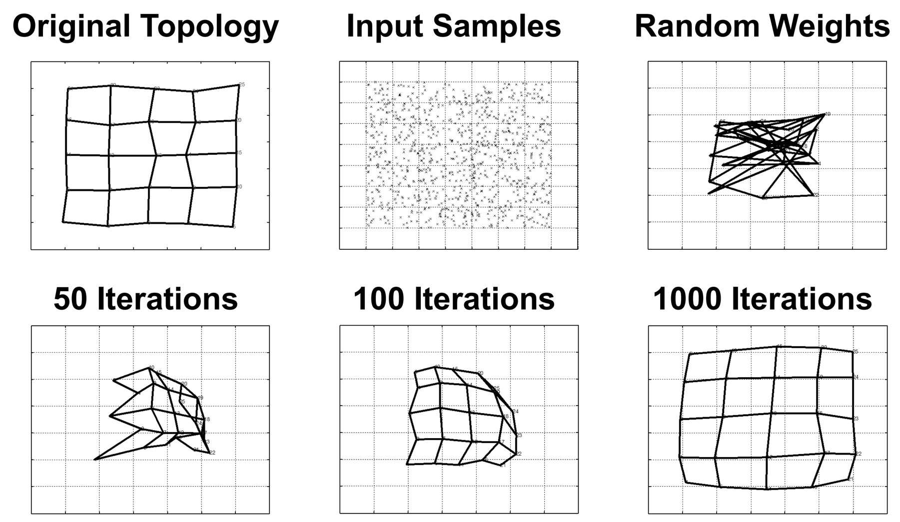
Node positions are computed by training a modified SOM where the weights represent 2D coordinates. Radio neighbors are mapped on adjacent position on the map.



Modified SOM used to compute the node locations

4. SOM Localization

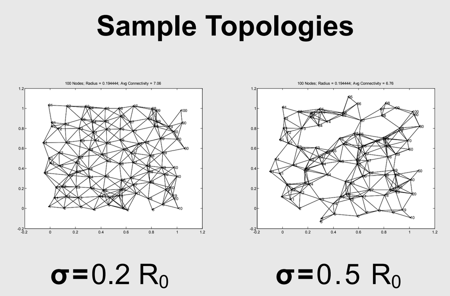
- Each weight vector represents the coordinates of a wireless node. The map is trained using random points drawn from an uniform 2D distribution.
- The hop-count distance between nodes is used to control the level of adaptation of the neurons.
- 1000 to 2000 iterations executed to achieve a good approximation of node positions.



Example: node positions computed by the SOM algorithm for a 25 node WSN with regular layout .

SIMULATION PARAMETERS:

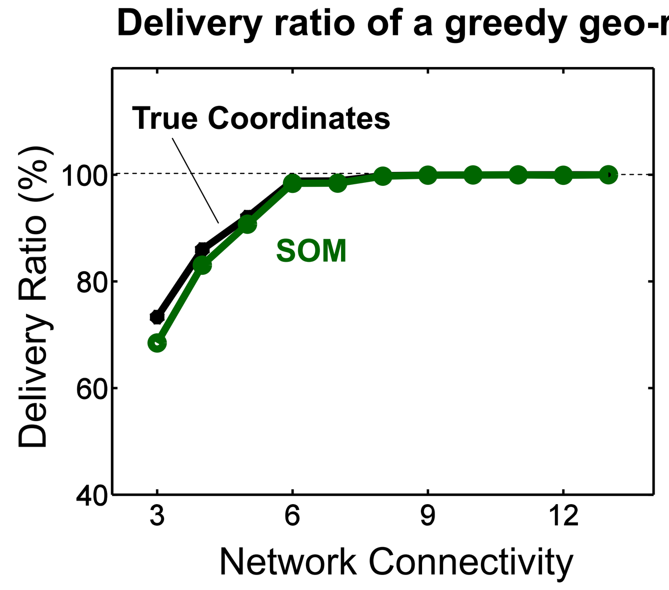
Number of Nodes: 16, 25, 36, 64, 81, 100
Communication Radius = {1.3, 1.6, 2.0, 2.2, 2.4}*R_0
Grid Perturbation σ = {0.1, 0.2, 0.3, 0.4, 0.5}*R_0
Degree of Irregularity - DOI = {0.0, 0.2, 0.4}
Total Number of Networks = ~ 30'000



5. Virtual Coordinates

- The SOM algorithm uses only connectivity information: sensors are localized without need of range measurements or anchor nodes. (**Range-Free, Anchor-Free** Localization).
- When no anchor nodes are used, results are arbitrarily scaled, translated, rotated or flipped. Node positions are expressed by **Virtual Coordinates**.
- Virtual Coordinates are useful to implement several network services, e.g. location-based queries, proximity-based service discovery and **Geographical Routing**.

Delivery ratio of a greedy geo-routing scheme



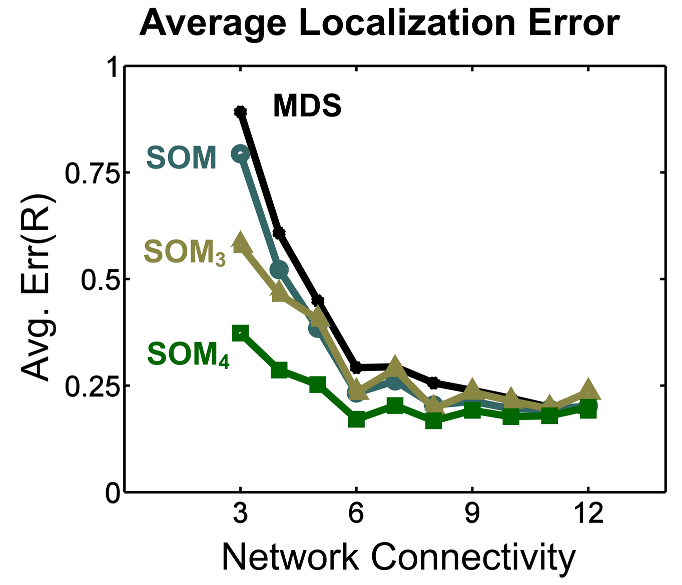
Simulation results show that the virtual coordinates generated by SOM are effective when used for geographical routing. Delivery ratio and path length* are close to the values obtained using the true coordinates.

*graph not shown

6. Absolute Coordinates

Relative maps can be converted into absolute maps using the positions of at least three anchor nodes.

Average Localization Error



In our simulations the positions of four anchor nodes on the perimeter of the map were used to transform the relative maps. The localization error was compared to the results obtained using Multi-Dimensional Scaling (MDS). The error is relative to the communication radius R.

Instead of using anchor node positions to convert virtual maps, the SOM algorithm can be modified to use this information **during** the computation.

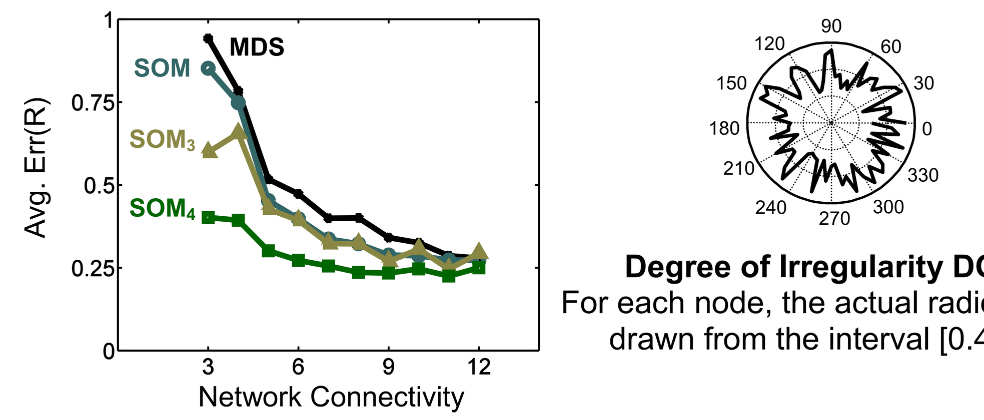
Neurons corresponding to anchor nodes are initialized with the true coordinates, which are never modified during the weights update.

SOM_3 and **SOM_4** use three and four anchor nodes. For networks with connectivity less than 10, the localization **error is reduced by 43%** with respect to MDS localization.

7. Irregular Networks

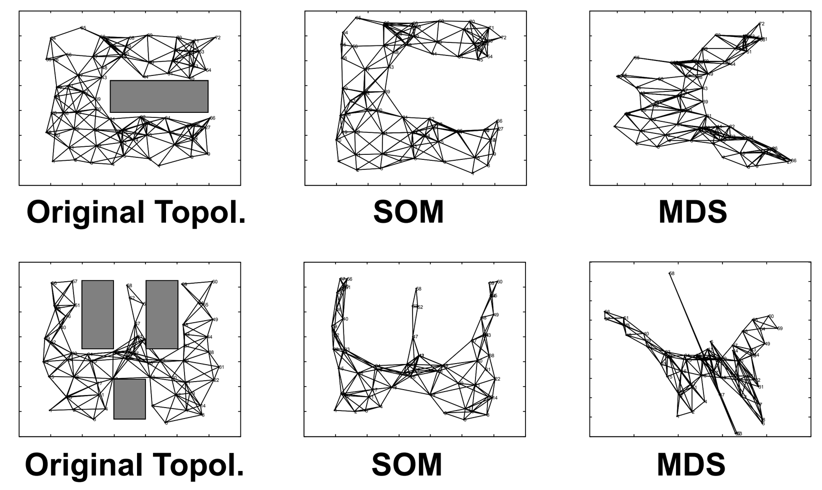
Simulation results in presence of irregular radio patterns or anisotropic layouts show that the SOM scheme maintains good localization results.

Localization error in presence of irregular radio patterns



Degree of Irregularity DOI = 0.4
For each node, the actual radio range was drawn from the interval [0.4 - 1.6] R

Sample results for anisotropic layouts (SOM and MDS)



SOM achieved an average localization error of 0.35R for networks with anisotropic layouts, with a **75% error reduction** with respect to MDS*.

* Result based on 400 simulated topologies.

8. Simulation Results

SOM Localization requires two steps:

1. Computing the hop-count distances (Dijkstra)
2. Training the SOM using multiple iterations.

Memory occupation and execution time on a TelosB board.

N. Nodes	Memory	Dijkstra	1000 iter.
36	0.42 KB	1 sec	62 sec
64	1.48 KB	6 sec	102 sec
100	3.42 KB	22 sec	156 sec

- MCU: MSP430 / 8 MHz
- Ram: 10KB / Flash: 48KB

9. Summary

- The SOM algorithm was used to implements a range-free, anchor-free localization scheme.
- Virtual coordinates generated by SOM are effective for location-aided routing.
- Accurate localization results for networks with low connectivity and in presence of irregular radio patterns or anisotropic layouts.
- SOM localization is computationally feasible.

ACKNOWLEDGEMENTS: this work was partly supported under contract IST – 508744 – IP EU Integrated Project GoodFood.