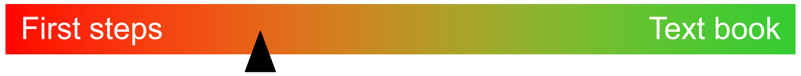
Positioning

Chapter 8

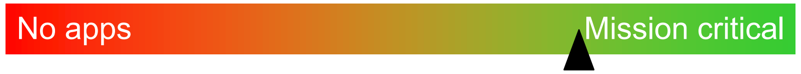
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Rating

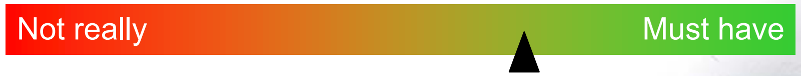
- Area maturity



- Practical importance



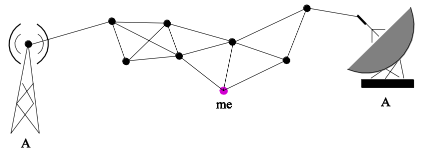
- Theoretical importance



Overview

Motivation

- Why positioning?
  - Sensor nodes without position information is often meaningless
  - Heavy and/or costly positioning hardware?
  - Geo-routing



- Why not GPS (or Galileo)?
  - Heavy, large, and expensive (as of yet)
  - Battery drain
  - Not indoors
  - Accuracy?
- Solution: equip small fraction with GPS (anchors)

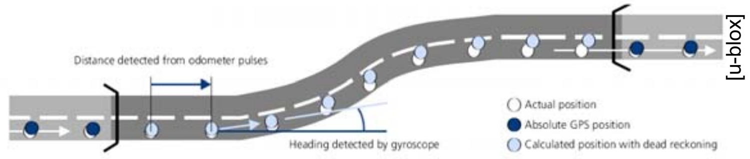
GPS

- A lot of recent progress, so attaching a GPS receiver to each sensor node becomes an alternative.
- Example, u-blox
  - footprint size: down to 4x4mm
  - Power supply: 1.8 - 4.8V
  - power consumption: 50 mW
  - power on: < 1 second
  - update rate: 4Hz
  - support for Galileo!

- So GPS is definitely becoming more attractive; however, some of the problems of GPS (indoors, accuracy, etc.) remain.

GPS Extensions

- GPS chips can be extended with other sensors such as gyroscope, direction indications, or tachometer pulses (in cars). With these add-ons, mobile devices get continued coverage indoors.


Distance detected from odometer pulses
Heading detected by gyroscope
Actual position
Absolute GPS position
Calculated position with dead reckoning
[u-blox]

- Affordable technology has a distance error of less than 5% per distance travelled, and a direction error of less than 3 degrees per minute.

Measurements

Distance estimation

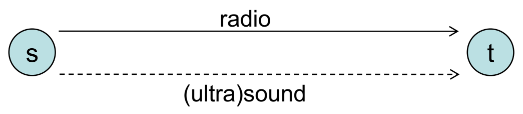
- Received Signal Strength Indicator (RSSI)
  - The further away, the weaker the received signal.
  - Mainly used for RF signals.
- Time of Arrival (ToA) or Time Difference of Arrival (TDoA)
  - Signal propagation time translates to distance.
  - RF, acoustic, infrared and ultrasound.

Angle estimation

- Angle of Arrival (AoA)
  - Determining the direction of propagation of a radio-frequency wave incident on an antenna array.
- Directional Antenna
- Special hardware, e.g., laser transmitter and receivers.

Example: Measuring distance with two radios

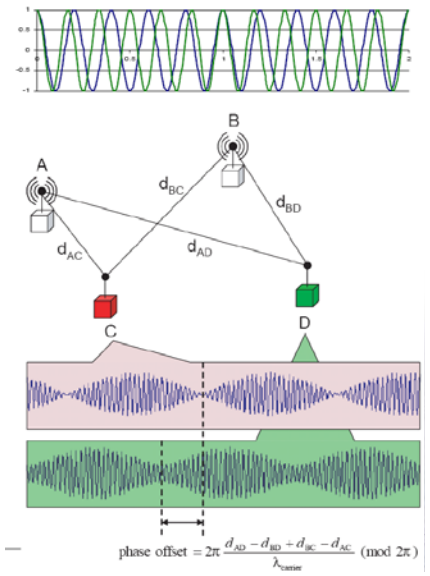
- Particularly interesting if the signal speed differs substantially, e.g. sound propagation is at about 331 m/s (depending on temperature, humidity, etc.), which is of course much less than the speed of light.



- If you have free sight you may achieve about a 1cm accuracy. But there are problems
  - (Ultra)sound does not travel far
  - For good results you need line of sight
  - You have to deal with reflections

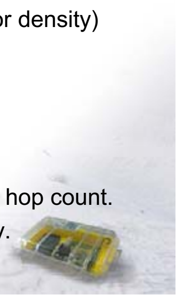
Inferometry

- Interferometry is the technique of superimposing (interfering) two or more waves, to detect differences between them
- Signals transmitted with a few hundred Hz difference at senders A and B will give different phase offsets at C and D. With that one can compute the total distance of the four points A, B, C, D.
- However, one needs to solve a linear equation system, and one needs very accurate time synchronization (μs order)


A
B
C
D
d_{BC}
d_{BD}
d_{AC}
d_{AD}
phase offset $= 2\pi \frac{d_{AD} - d_{BD} + d_{BC} - d_{AC}}{\lambda_{carrier}} \pmod{2\pi}$

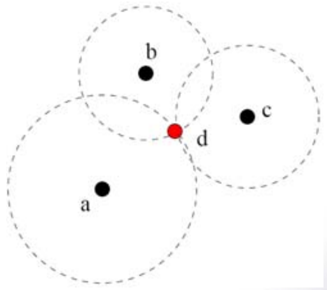
Positioning (a.k.a. Localization)

- Task: Given distance or angle measurements or mere connectivity information, find the locations of the sensors.
- Anchor-based
  - Some nodes know their locations, either by a GPS or as pre-specified.
- Anchor-free
  - Relative location only. Sometimes called virtual coordinates.
  - Theoretically cleaner model (less parameters, such as anchor density)
- Range-based
  - Use range information (distance or angle estimation).
- Range-free
  - No distance estimation, use connectivity information such as hop count.
  - It was shown that bad measurements don't help a lot anyway.



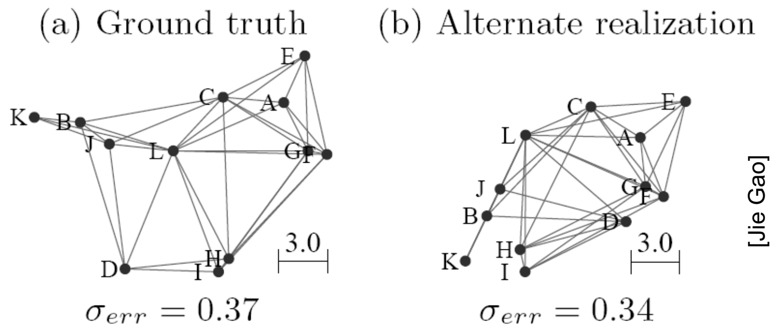
Trilateration and Triangulation

- Use geometry, measure the distances/angles to three anchors.
- Trilateration: use distances
  - Global Positioning System (GPS)
- Triangulation: use angles
  - Some cell phone systems
- How to deal with inaccurate measurements?
  - Least squares type of approach
  - What about strictly more than 3 (inaccurate) measurements?

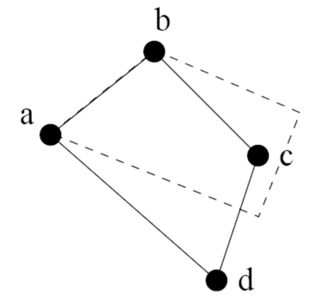


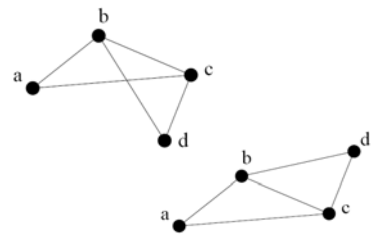
Ambiguity Problems

- Same distances, different realization.



Continuous deformation, flips, etc.



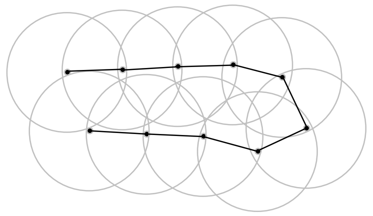


[Jie Gao]

- Rigidity theory: Given a set of rigid bars connected by hinges, rigidity theory studies whether you can move them continuously.

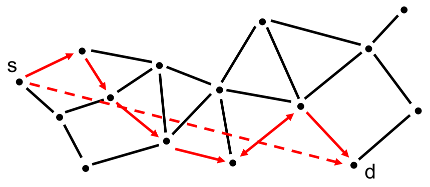
Simple hop-based algorithms

- Algorithm
  - Get graph distance h to anchor(s)
  - Intersect circles around anchors
    - radius = distance to anchor
  - Choose point such that maximum error is minimal
    - Find enclosing circle (ball) of minimal radius
    - Center is calculated location
- In higher dimensions: $1 < d \leq h$
  - Rule of thumb: Sparse graph
    → bad performance



How about no anchors at all...?

- In absence of anchors...
  → ...nodes are clueless about real coordinates.
- For many applications, real coordinates are not necessary
  → Virtual coordinates are sufficient
  → Geometric Routing requires only virtual coordinates
    - Require no routing tables
    - Resource-frugal and scalable



Virtual Coordinates

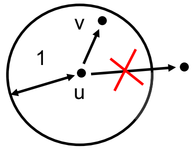
- Idea:
  Close-by nodes have similar coordinates
  Distant nodes have very different coordinates

→ Similar coordinates imply physical proximity!

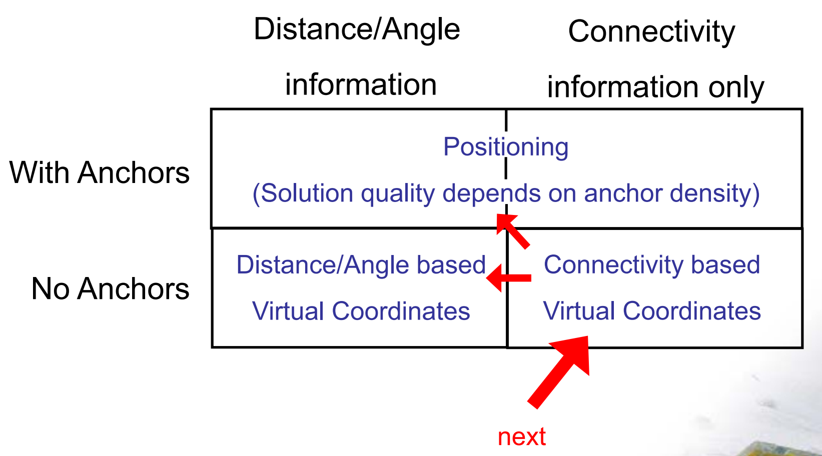
- Applications
  - Geometric Routing
  - Locality-sensitive queries
  - Obtaining meta information on the network
  - Anycast services („*Which of the service nodes is closest to me?*")
  - Outside the sensor network domain: e.g., Internet mapping

Model

- Unit Disk Graph (UDG) to model wireless multi-hop network
  - Two nodes can communicate iff Euclidean distance is at most 1
- Sensor nodes may not be capable of
  - Sensing directions to neighbors
  - Measuring distances to neighbors
- Goal: Derive topologically correct coordinate information from connectivity information only.
  - Even the simplest nodes can derive connectivity information



Context



	Distance/Angle information	Connectivity information only
With Anchors	Positioning (Solution quality depends on anchor density)	
No Anchors	Distance/Angle based Virtual Coordinates	Connectivity based Virtual Coordinates

Virtual Coordinates ⟷ UDG Embedding

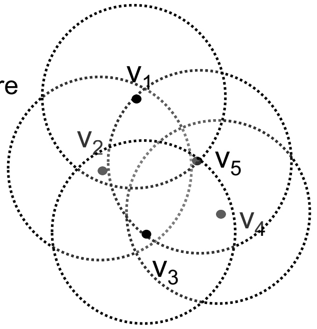
- Given the connectivity information for each node...

v_1	v_2	v_3	v_4	v_5
v_2	v_1	v_2	v_3	v_1
v_5	v_3	v_4	v_5	v_3
		v_5		v_4

...and knowing the underlying graph is a UDG...

...find a UDG embedding in the plane such that all connectivity requirements are fulfilled! (→ Find a realization of a UDG)

This problem is NP-hard!
(Simple reduction to *UDG-recognition* problem, which is NP-hard)
[Breu, Kirkpatrick, Comp.Geom.Theory 1998]



UDG Approximation – Quality of Embedding

- Finding an exact realization of a UDG is NP-hard.
  → Find an embedding r(G) which approximates a realization.
- Particularly,
  → Map adjacent vertices (edges) to points which are close together.
  → Map non-adjacent vertices („non-edges“) to far apart points.
- Define quality of embedding q(r(G)) as:

Ratio between longest edge to shortest non-edge in the embedding.

Let $\rho(u,v)$ be the distance between points u and v in the embedding.

$$q(r(G)) := \frac{\max_{\{u,v\}\in E}\rho(u,v)}{\min_{\{u',v'\}\notin E}\rho(u',v')}$$

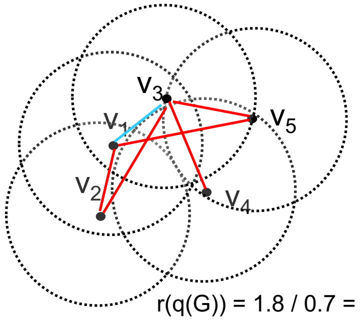
UDG Approximation

- For each UDG G, there exists an embedding r(G), such that, q(r(G)) $\leq$ 1. (a realization of G)

$$\boxed{q(r(G)) := \frac{\max_{\{u,v\}\in E} \rho(u,v)}{\min_{\{u',v'\}\notin E} \rho(u',v')}}$$

- Finding such an embedding is NP-hard
- An algorithm ALG achieves approximation ratio α if for all unit disk graphs G, q(r_{ALG}(G))$\leq \alpha$.
- Example:

v_1	v_2	v_3	v_4	v_5
v_2	v_1	v_2	v_3	v_1
v_5	v_3	v_4	v_5	v_3
		v_5		v_4



r(q(G)) = 1.8 / 0.7 = 2.6

Some Results

- There are a few virtual coordinates algorithms
  Almost all of them evaluated only by simulation on random graphs
- In fact there are very few provable approximation algorithms

There is an algorithm which achieves an approximation ratio of $O(\log^{2.5} n \sqrt{\log\log n})$, n being the number of nodes in G.

Plus there are lower bounds on the approximability.

There is no algorithm with approximation ratio better than $\sqrt{3/2} - \epsilon$ unless P = NP.

Approximation Algorithm: Overview

- Four major steps

1. Compute metric on MIS of input graph → Spreading constraints
   (Key conceptual difference to previous approaches!)
2. Volume-respecting, high dimensional embedding
3. Random projection to 2D
4. Final embedding

UDG Graph G with MIS M.

↓

Approximate pairwise distances between nodes such that, MIS nodes are neatly spread out.

↓

Volume respecting embedding of nodes in $\mathbb{R}^n$ with small distortion.

↓

Nodes spread out fairly well in $\mathbb{R}^2$.

↓

Final *embedding of G in* $\mathbb{R}^2$.

Lower Bound: Quasi Unit Disk Graph

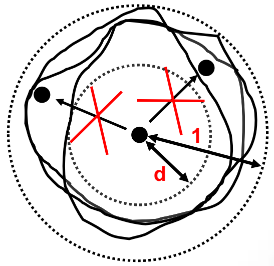
- Definition Quasi Unit Disk Graph:

Let V$\in \mathbb{R}^2$, and d $\in$ [0,1]. The symmetric Euclidean graph G=(V,E), such that for any pair u,v $\in$ V

dist(u,v) $\leq$ d $\Rightarrow$ {u,v} $\in$ E

dist(u,v) > 1 $\Rightarrow$ {u,v} $\notin$E

is called *d-quasi unit disk graph*.



Note that between d and 1, the existence of an edge is unspecified.

Reduction

- We want to show that finding an embedding with $q(r(G)) \le \sqrt{3/2} - \epsilon$, where ε goes to 0 for n → ∞ is NP-hard.
- We prove an equivalent statement:

> Given a unit disk graph G=(V,E), it is NP-hard to find a realization of G as a d-quasi unit disk graph with $d \ge \sqrt{2/3} + \epsilon$, where ε tends to 0 for n→∞.

→ Even when allowing non-edges to be smaller than 1, embedding a unit disk graph remains NP-hard!
→ It follows that finding an approximation ratio better than $\sqrt{3/2} - \epsilon$ is also NP-hard.

Reduction

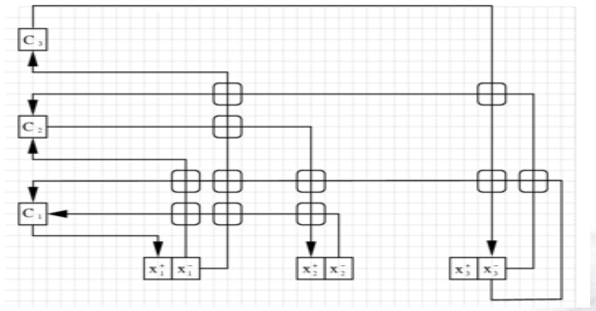
- Reduction from 3-SAT (each variable appears in at most 3 clauses)
- Given a instance C of this 3-SAT, we give a polynomial time construction of $G_C=(V_C, E_C)$ such that the following holds:

> – C is satisfiable ⇒ G_C is realizable as a unit disk graph
> – C is not satisfiable ⇒ G_C is not realizable as a d-quasi unit disk graph with $d \ge \sqrt{2/3} + \epsilon$

- Unless P=NP, there is no approximation algorithm with approximation ratio better than $\sqrt{3/2} - \epsilon$.

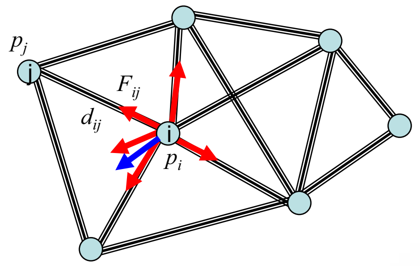
Proof idea

- Construct a grid drawing of the SAT instance.
- Grid drawing is *orientable* iff SAT instance is satisfiable.
- Grid components (clauses, literals, wires, crossings,...) are composed of nodes → Graph G_C.
- G_C is realizable as a d-quasi unit disk graph with $d \ge \sqrt{2/3} + \epsilon$ iff grid drawing is orientable.



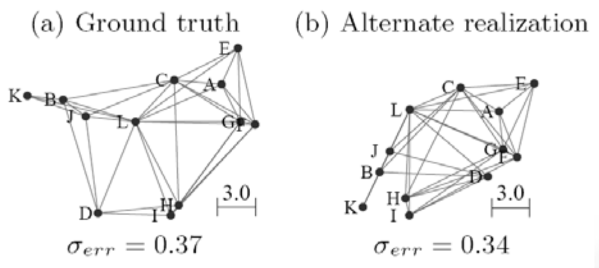
Heuristics: Spring embedder

- Nodes are “masses”, edges are “springs”.
- Length of the spring equals the distance measurement.
- Springs put forces to the nodes, nodes move, until stabilization.
- Force: $F_{ij} = d_{ij} - r_{ij}$, along the direction $p_i p_j$.
- Total force on n_i: $F_i = \Sigma F_{ij}$.
- Move the node n_i by a small distance (proportional to F_i).



Spring Embedder Discussion

- Problems:
  - may deadlock in local minimum
  - may never converge/stabilize (e.g. just two nodes)
- Solution: Need to start from a reasonably good initial estimation.



[Jie Gao]

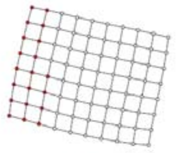
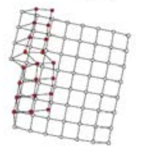
Heuristics: Priyantha et al.

N.B. Priyantha, H. Balakrishnan, E. Demaine, S. Teller: **Anchor-Free Distributed Localization in Sensor Networks**, *SenSys*, 2003.

iterative process minimizes the layout energy

$$E(p) = \sum_{\{i,j\}\in E} \left(\|p_i - p_j\| - \ell_{ij}\right)^2$$

- fact: layouts can have *foldovers* without violating the distance constraints
- problem: optimization can converge to such a local optimum
- solution: find a good initial layout *fold-free* $\rightarrow$ already close to the global optimum (="real layout")

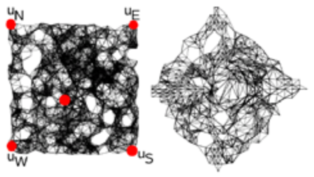



[Fleischer & Pich]

Continued

Phase 1: compute initial layout

- determine periphery nodes u_N, u_S, u_W, u_E
- determine central node u_C
- use polar coordinates



$$\rho_v = d(v, u_C) \quad \theta_v = \arctan\left(\frac{d(v,u_N) - d(v,u_S)}{d(v,u_W) - d(v,u_E)}\right)$$

as positions of node v

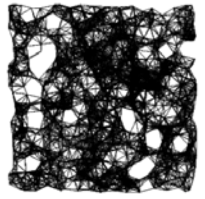
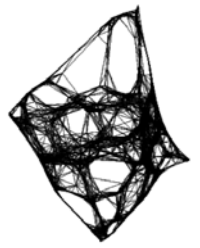
Phase 2: Spring Embedder

[Fleischer & Pich]

Heuristics: Gotsman et al.

C. Gotsman, Y. Koren [5]. **Distributed Graph Layout for Sensor Networks**, *GD*, 2004.

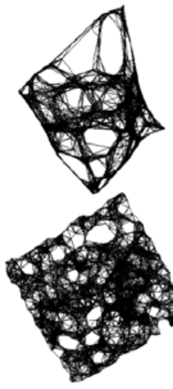
- initial placement: spread sensors
  $$\frac{\sum_{\{i,j\}\in E} \exp(-\ell_{ij})\|p_i - p_j\|^2}{\sum_{i<j}\|p_i - p_j\|^2} \rightarrow \min$$
- linear algebra:
  minimized by second highest eigenvector v_2 of A where
  $$a_{ij} = -\frac{\exp(-\ell_{ij})}{\sum_{j:\{i,j\}\in E}\exp(-\ell_{ij})}$$
  $$a_{ii} = 1$$
- $x, Ax, A^2x, A^3x, \ldots$ converges to v_2
- $x_i \leftarrow \frac{1}{2}\left(x_i + \frac{\sum_{j:\{i,j\}\in E}\exp(-\ell_{ij}x_j)}{\sum_{j:\{i,j\}\in E}\exp(-\ell_{ij})}\right)$
- compute third eigenvector v_3, use v_2, v_3 as coordinates

[Fleischer & Pich]

Continued

- distributed optimization (spring model)
- alternative: *majorization*
- compute sequence of layouts $p^{(0)}, p^{(1)}, p^{(2)}, \ldots$ with $E(p^{(0)}) \geq E(p^{(1)}) \geq E(p^{(2)}) \geq \ldots$
  - solve linear equation $L^{(t+1)} p^{(t+1)} = L^{(t)} p^{(t)}$ in distributed manner

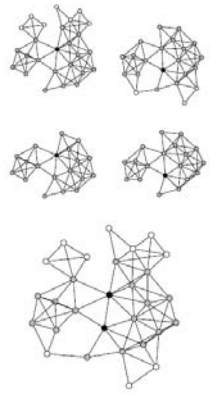


[Fleischer & Pich]

Heuristics: Shang et al.

Y. Shang, W. Ruml [7].
Improved MDS-based Localization, *IEEE Infocom*, 2004.

- compute a local map for each node (local MDS of the 2-hop neighborhood)
- merge local map patches into a global map (use incremental or binary-tree strategy)
- apply distributed optimization to the result

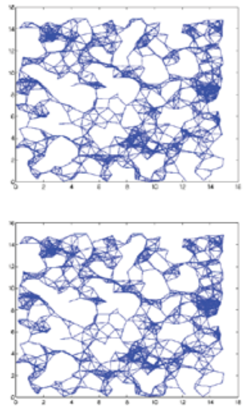


[Fleischer & Pich]

Heuristics: Bruck et al.

J. Bruck, J. Gao, A. Jiang [8]. **Localization and Routing in Sensor Networks by Local Angle Information**, *Mobile Ad Hoc Networking & Computing*, 2005.

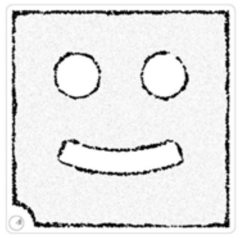
- Choose an edge e as x-axis to obtain absolute angles.
- Form an LP whose variables are the edge lengths $\ell(e)$.
- For all edges $0 \leq \ell(e) \leq 1$.
- For any cycle $e_1, \ldots, e_p$:
  $\sum_{i=1}^{p} \ell(e_i) \cos\theta_i = 0$ and
  $\sum_{i=1}^{p} \ell(e_i) \sin\theta_i = 0$.
- Non-adjacent node pair constraints.
- Crossing-edge constraints.

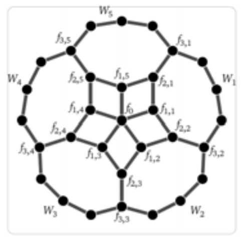


[Fleischer & Pich]

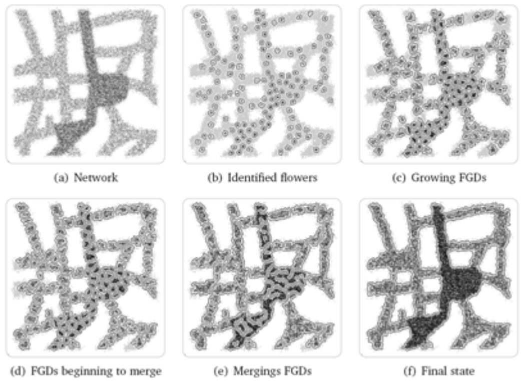
Boundary recognition

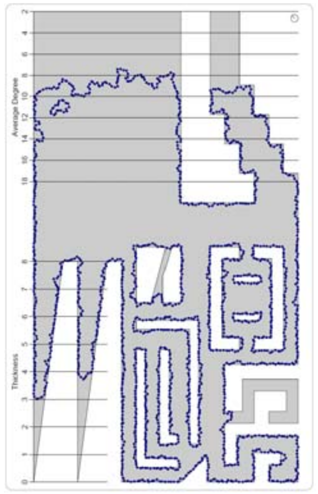
- Related problem, given a connectivity graph, what is the boundary?
- So far heuristics only, one heuristic uses the idea of a independent nodes; specifically, an interior (non-boundary) node sees enough (e.g. 5) independent neighbors which in turn have a ring around them; these are called “flowers”. Flowers can be grown and connected to compute the boundary.
- However, this is only a heuristic, and does not always work…





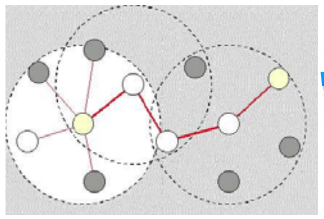
Boundary recognition





Practical lessons

Theory



Practice

RSSI in sensor networks: good, but not for “reasonable” localization

For exact indoor localization
- Buy special hardware (e.g., UWB)
- Place huge amount of short range anchors for single-hop localization

Open problem

- One tough open problem of this chapter obviously is the UDG embedding problem: Given the adjacency matrix of a unit disk graph, find positions for all nodes in the Euclidean plane such that the ratio between the maximum distance between any two adjacent nodes and the minimum distance between any two non-adjacent nodes is as small as possible.
- There is a large gap between the best known lower bound, which is a constant, and the polylogarithmic upper bound. It is a challenging task to either come up with a better approximation algorithm or prove a stronger (non-constant) lower bound. Once we understand this better, we can try networks with anchors, or with (approximate) distance/angle information.
- Generally, beyond GPS this area is in its infancy.