Rigidity and Localization: An Optimization Perspective

Anthony Man-Cho So

Department of Systems Engineering & Engineering Management

The Chinese University of Hong Kong

Operations Research Seminar

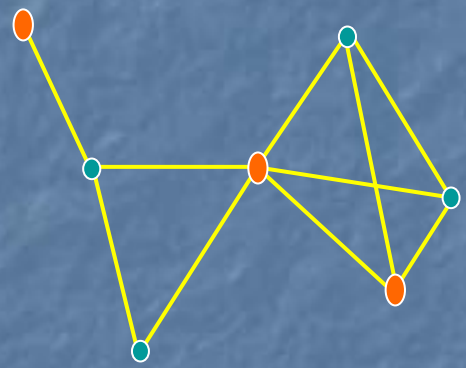
Department of Management Science & Engineering

Stanford University

15 March 2010

Sensor Network Localization

- Given:
  - set of sensors V_s and anchors V_a
  - set of sensor-sensor edges E_{ss}
  - set of sensor-anchor edges E_{sa}
  - edge weights $\{d_{ij} \geq 0 : (i, j) \in E_{ss}\}$ and $\{\bar{d}_{ij} \geq 0 : (i, j) \in E_{sa}\}$
  - an integer d
- Goal:
  - place the vertices of G in R^d so that their coordinates satisfy the anchor and distance constraints



Background

- The problem is computationally intractable (Saxe 1979, Aspnes et al. 2004) for $d \geq 1$.
  - This should be contrasted with the complexity of determining whether a generic instance has a unique realization in R^d.
- Many heuristics have been proposed:
  - global optimization
  - trilateration
  - ad-hoc approaches
  - ...

Background

- Much recent interest in convex relaxation approaches (initiated by Doherty et al. 2001, Biswas and Ye 2004).
  - Good computational and theoretical properties.

The Biswas-Ye SDP Model

- The problem can be formulated as follows:

$$\|x_i - x_j\|^2 = d_{ij}^2 \qquad (i, j) \in E_{ss}$$

$$\|a_k - x_j\|^2 = \bar{d}_{kj}^2 \qquad (k, j) \in E_{sa}$$

$$x_i \in R^d$$

$\{a_k\}$ are the positions of "anchors".

- The above system is non-convex and generally intractable. To get something more tractable, we can consider a convex relaxation.

Getting a Convex Relaxation

- Step 1: Linearize

$$\|x_i - x_j\|^2 = \underbrace{x_i^T x_i}_{Y_{ii}} - 2\underbrace{x_i^T x_j}_{Y_{ij}} + \underbrace{x_j^T x_j}_{Y_{jj}}$$

$$\|a_k - x_j\|^2 = a_k^T a_k - 2a_k^T x_j + \underbrace{x_j^T x_j}_{Y_{jj}}$$

- Step 2: Tighten

$$Y \geq X^T X \Leftrightarrow Z = \begin{bmatrix} I & X \\ X^T & Y \end{bmatrix} \geq 0$$

The SDP Relaxation

- Use linear algebra trick and put things together:

$$\begin{aligned}
&\left(0; e_i - e_j\right)\left(0; e_i - e_j\right)^T \bullet Z = d_{ij}^2 && (i,j) \in E_{ss} \\
&\left(a_k; -e_j\right)\left(a_k; -e_j\right)^T \bullet Z = \bar{d}_{kj}^2 && (k,j) \in E_{sa} \\
&Z \geq 0;\ Z_{1:d,1:d} = I_d
\end{aligned}$$

- This is an instance of semidefinite programming (SDP), which can be solved (to any arbitrary accuracy) in polynomial time.
- An important and interesting question is: when is the relaxation *exact*?

The Biswas-Ye SDP Model

- The Biswas-Ye SDP has some nice properties:
  - (Biswas and Ye 2004) If Z has rank r, then we can extract from it a r-d realization.
    - Thus, if rank(Z)=d, then we solve the original problem (i.e. relaxation is exact).
  - (S. and Ye 2005) The relaxation is exact iff the input satisfies the following uniqueness property:

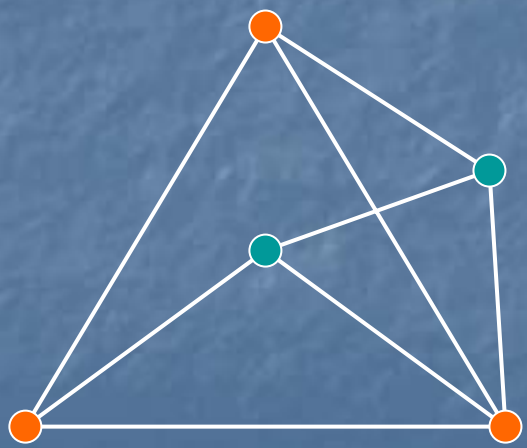
    Unique d-Realizability: The input has a unique realization in R^d, and does not have any non-trivial realization in R^h for $h>d$.
  - (S. and Ye 2006) The optimal dual multipliers can be interpreted as tension on the edges.

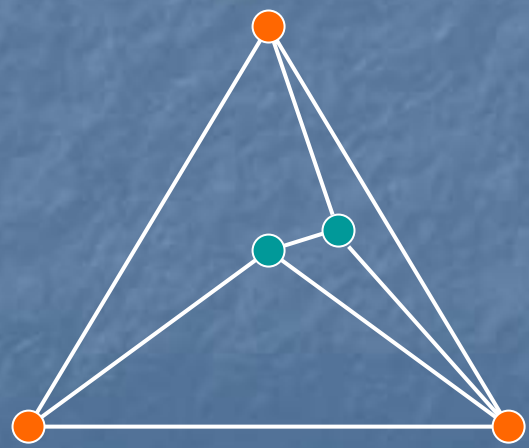
Unique d-Realizability

- The notion of unique d-realizability raises two immediate questions:
  - Does such an instance exist?
  - Where does it fit in the theory of rigidity?

Existence of Unique d-Realizable Instances

- It turns out that there are many uniquely d-realizable instances, and the graphs need not be dense (more on this later).
- The notion depends on both the combinatorial and geometric properties of the input.





Unique d-Realizability and Rigidity

- The notion of unique d-realizability motivates the following rigidity-theoretic definition:

  Definition: Let $G=(V,E)$ be an l-vertex graph, and $p=(p_1,\ldots,p_l)$ be its realization in R^d. We say that (G,p) is universally rigid (UR) in R^d if it is the unique (up to congruence) realization in any Euclidean space.

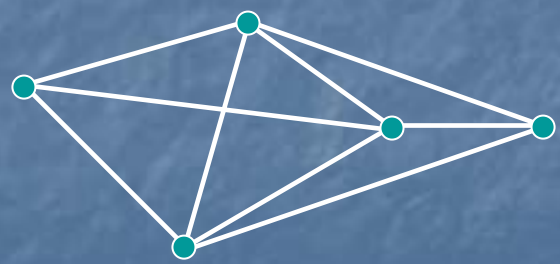
  If (G,p) is universally rigid in R^d for all generic realizations p, then we say that G is generically universally rigid (GUR) in R^d.

Unique d-Realizability and Rigidity

- It can be shown (Zhu, S., Ye 2010) that the notions of unique d-realizability and UR in R^d are in some sense equivalent.
- Also, GUR is a combinatorial notion – it only depends on the graph.
  - A combinatorial version of unique d-realizability.
- The next question is, is there any GUR graph?

Construction of GUR Graphs

- Consider the so-called d-trilateration graphs (e.g. Eren et al. 2004):
  - K_{d+1} is a d-trilateration graph.
  - A graph G with $n+1$ vertices, where $n>d$, is a d-trilateration graph if it is obtained by adding a vertex to some n-vertex d-trilateration graph G' and connecting it to at least $(d+1)$ vertices of G'.



$d=2$

Construction of GUR Graphs

- Theorem (Zhu, S., Ye 2010; cf. S. 2007): d-trilateration graphs are GUR in R^d.
- Note that a minimal d-trilateration graph with $n+d+1$ vertices has only $O(dn)$ edges.
  - In particular, they are sparse.
- Research Question: Are there other GUR graphs?
  - (Gortler, Thurston 2010): Let G be GGR in R^d. Then, G^2 is GUR in R^d.

Further Exploitation of the SDP Model

- So far we are only considering an SDP feasibility problem.
- We can do more with a suitable objective!
- This is motivated by ideas in tensegrity theory.
- In return, we show that SDP duality theory is a powerful proof technique for problems in tensegrity theory.

A Heuristic

- Recall our goal: to find a low-dimensional realization.
- Consider the graph G:

- For any $k=1,\ldots,n-1$, we can realize G in R^k. However, if $\|v_1 - v_n\|$ is to be maximized, then we can only realize G in R.

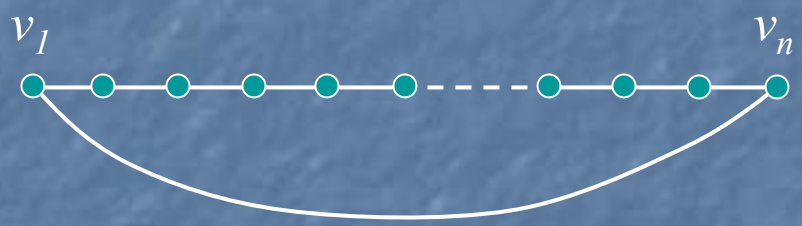
Certificate of Low-Dimensionality

- So it seems like a good idea to stretch a non-adjacent pair...
- But how do we formalize this observation?

Existence of a Non-Zero Equilibrium Stress

Equilibrium Stress

- Consider the graph G':



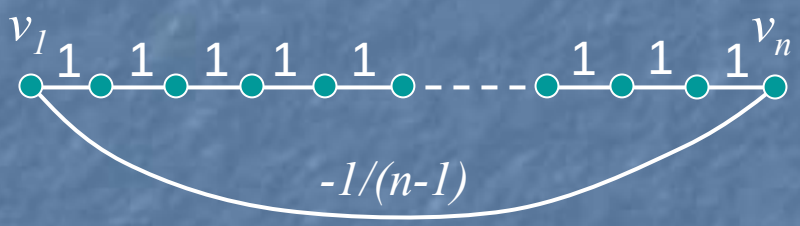
- The maximization induces a set of multipliers θ_{ij} on the edges of G', not all zero, such that:

$$\sum_{j:(i,j)\in E} \theta_{ij}\left(v_j - v_i\right) = \vec{0} \quad \text{for all } i$$

- $\{\theta_{ij}\}$ is called an equilibrium stress for G'.

Equilibrium Stress

- Back to the previous example:



- Observe that the equilibrium stress depends both on the graph (combinatorics) and on the positions of the vertices (geometry).

Why Equilibrium Stress?

- So why is a non-zero equilibrium stress useful?

  (Local) Dimension Reduction

- In particular, the vectors $\{v_j\text{-}v_i\}$ around each vertex v_i are linearly dependent.

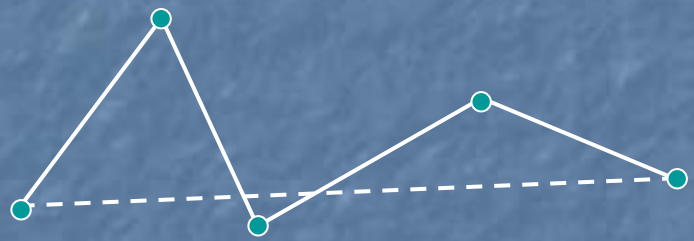
$$\sum_{j:(i,j)\in E} \theta_{ij}\left(v_j - v_i\right) = \vec{0} \quad \text{for all } i$$

Existence of Equilibrium Stress

- A natural question is, given a graph and one of its realizations in R^k, does it always admit a non-zero equilibrium stress?

Existence of Equilibrium Stress

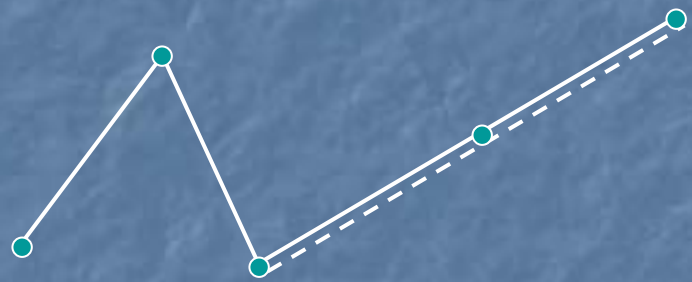
- No! For example,



- However, by stretching a non-adjacent pair of vertices (“adding a strut”), we can always obtain a non-zero equilibrium stress.

Existence of Equilibrium Stress

- More precisely, any realization that maximizes the strut length induces a non-zero equilibrium stress on that realization.



An Existence Proof

- This theorem is proven by Robert Connelly and Maria Sloughter (now Maria Belk) in 2004.
- However, the proof is not constructive (uses the Inverse Function Theorem).
- In particular, it does not provide:
  - the maximizing realization, and
  - its associated non-zero equilibrium stress.

A Constructive Proof via SDP

- The Connelly-Sloughter theorem can be proven (and sharpened) using SDP duality theory (S., Ye 2006).
- As a result, we have an efficient algorithm that computes both a realization and a non-zero equilibrium stress on that realization.
- The proof uses complementarity between the optimal primal and dual solutions.
- Research Question: Rank Reduction via objective design?

Convex Relaxation is Great, or?

- Note that the Biswas-Ye SDP has
  - $(d+n)(d+n+1)/2$ variables and
  - $d(d+1)/2+|E_{ss}|+|E_{sa}|$ constraints.
- Thus, it is time-consuming to solve the SDP as is.

Existing Speedup Approaches

- Much recent effort has focused on speeding up the computation time of convex relaxation-based approaches:
  - ad-hoc constraint removal (e.g. Biswas et al. 2006, 2008, Carter et al. 2006, Wang et al. 2008, Kim et al. 2009)
  - exploit sparsity in existing formulations (e.g. Kim et al. 2009)
  - develop alternative, less tight relaxations (e.g. Tseng 2007, Wang et al. 2008)
  - clique reduction (Krislock and Wolkowicz 2009)

Issues with Existing Approaches

- Ad-hoc constraint removal and alternative relaxations:
  - do not have the accuracy of the original SDP
- Sparsity-exploiting techniques:
  - derived from general-purpose speedup techniques
  - do not exploit the structure of the sensor network localization problem
- Clique Reduction:
  - local buildup, may get tripped up by local ambiguities

GUR Graphs as a Speedup Tool

- Exploit the theoretical properties of the Biswas-Ye SDP and GUR graphs and turn them into an edge sparsification procedure.
- Such a procedure can
  - provably preserve the localization properties of the input, and
  - in general reduce the number of constraints in the SDP.

High-Level Idea

- Given an arbitrary sensor network localization instance, identify d-trilateration subgraphs and sparsify those subgraphs accordingly.

Identify d-trilateration Subgraphs

- Recall the definition of d-trilateration graphs:
  - K_{d+1} is a d-trilateration graph.
  - A graph G with $n+1$ vertices, where $n>d$, is a d-trilateration graph if it is obtained by adding a vertex to some n-vertex d-trilateration graph G' and connecting it to at least ($d+1$) vertices of G'.

Identify d-trilateration Subgraphs

- Now, given an arbitrary localization instance, we can
  - decompose the input graph into a number of d-trilateration subgraphs whenever possible, and
  - remove any redundant edges in those subgraphs.
- Theorem (Zhu, S., Ye 2010): For $d=2$, there exists an $O(n^3)$ algorithm that decompose the input graph as $G = G_1 + \cdots + G_k$, where
  - $G_1,\ldots,G_{k-1}$ are minimal 2-trilateration graphs, and
  - G_k is either a minimal 2-trilateration graph, or is a triangle-free graph.

Identify d-trilateration Subgraphs

- In particular, the number of edges in the sparsified graph can be bounded:
  - $O(k|V|)$ if G_k is a minimal 2-trilateration graph
  - $O(k|V|+|V_k|^2)$ if G_k is a triangle-free graph

Identify d-trilateration Subgraphs

- Besides improving computational efficiency, we also retain the accuracy of the relaxation:

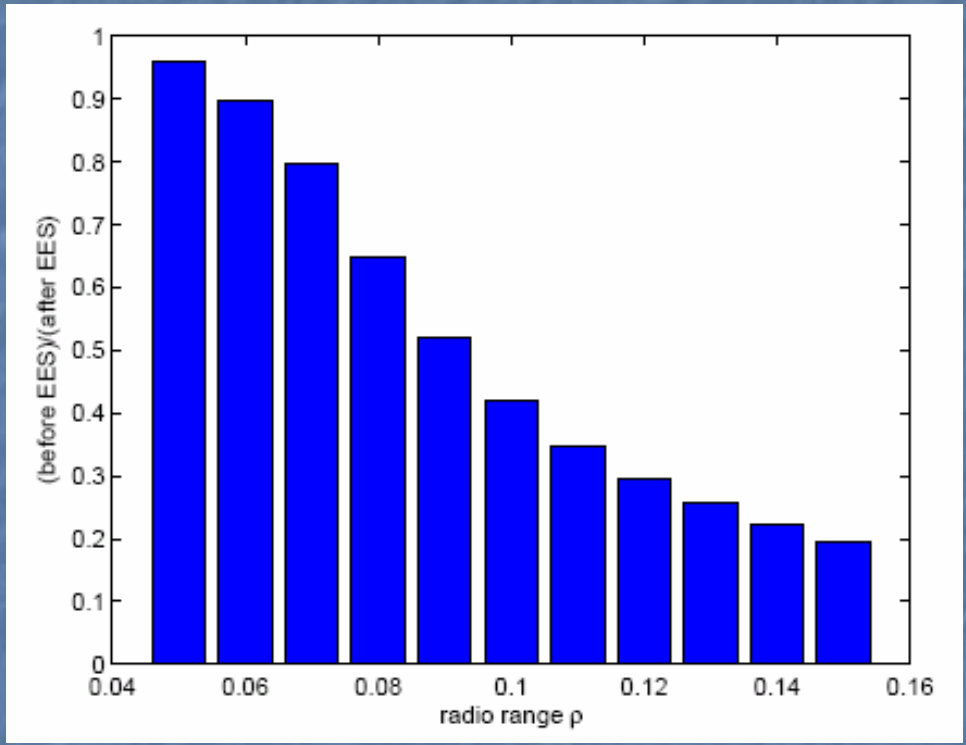
  Theorem (Zhu, S., Ye 2010): Consider the localization instance $\left(G, \{d_{ij}\}, \{\overline{d}_{kj}\}, d\right)$. Let G' be the sparsified graph. If p is a generic realization of G', then it is also a generic realization of G.

  Theorem (Zhu, S., Ye 2010): Let $\left(G, \{d_{ij}\}, \{\overline{d}_{kj}\}, d\right)$ be a uniquely d-realizable instance, and let G' be the sparsified graph. Then, $\left(G', \{d_{ij}\}, \{\overline{d}_{kj}\}, d\right)$ is also uniquely d-realizable.

Simulation Results

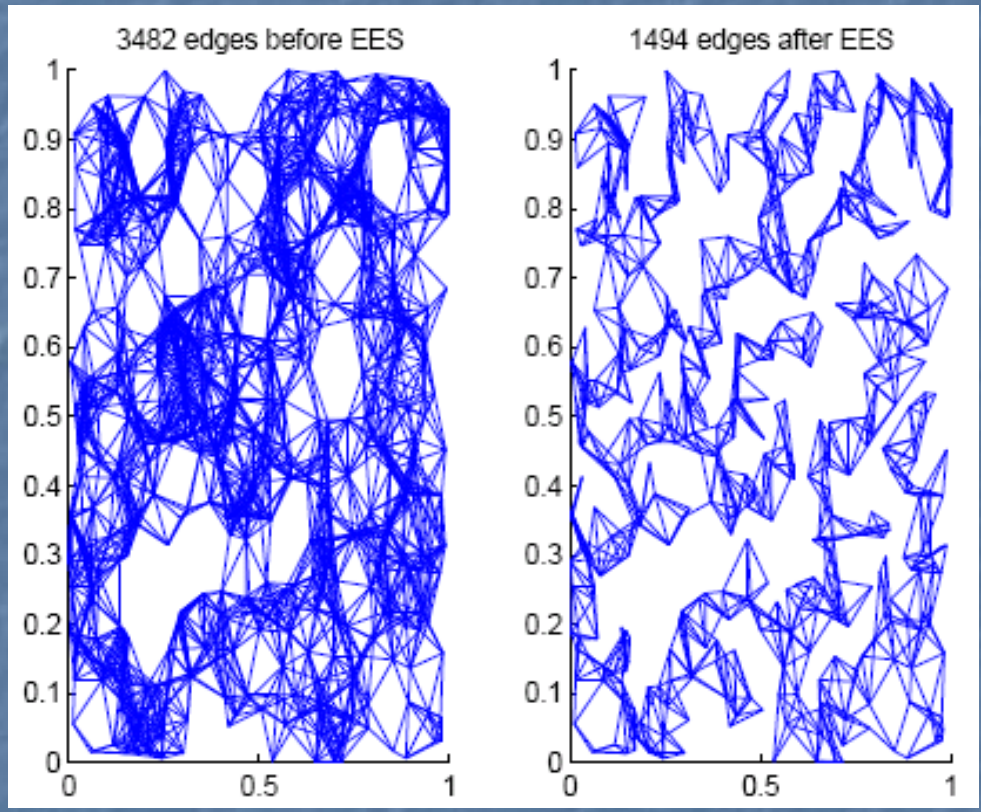
- Setup:
  - unit-disk graph model
  - random placement of 500 sensors over unit square

Effectiveness of Sparsification Procedure



The ratios of the number of edges before and after applying the sparsification procedure.

Effectiveness of Sparsification Procedure



Effect of the sparsification procedure on a randomly generated 500-node instance.

Efficiency Gain of the Sparsification Procedure

n	FSDP	EES–FSDP	SSDP	EES–SSDP
100	4.8	3.8	5.9	3.8
200	24.0	14.4	23.9	8.7
400	262.8	118.2	110.9	27.6
800	2439.3	1116.9	674.7	115.1
1600	*	*	*	639.1

FSDP = original Biswas-Ye formulation (Biswas and Ye 2004)

SSDP = sparse SDP formulation of Kim et al. (Kim et al. 2009)

EES-FSDP = FSDP with edge sparsification

EES-SSDP = SSDP with edge sparsification

* = out of memory

Conclusion

- Optimization theory offers some new tools for tackling problems in rigidity theory and localization.
- It would be interesting to further exploit the connection.

Thank You!