Category based on Computational Organization

- second, optimize a global metric over the network
- relaxation based distributed algorithm is to use a coarse algorithm to roughly localize nodes and followed by refinement step involving node position adjustment to optimize a local error metric
- algorithms hope to approximate the optimal solution to a network-wide metric that is the sum of the local error metric at each of the nodes
- coordinate system stitching approach optimizing a network wide metric in a distributed manner
- the network is divided into small overlapping sub-regions, each of which creates an optimal local map
- the sub-regions use a peer-to-peer process to merge their local maps into a single global map; this global map approximates the global optimum map

Centralized Algorithms

- Two types of processing: semidefinite programming (SDP) and multidimensional scaling (MDS)
- SDP approach to localization was pioneered by Doherty et al.
- geometric constraints between nodes are represented as linear matrix inequalities (LMIs)
- once all the constraints in the network are expressed in this form, the LMIs can be combined to form a single semidefinite program
- this is solved to produce a bounding region for each node, which Doherty et al simplify to be a bounding box
- unfortunately, not all geometric constraints can be expressed as LMIs
- only constraints that form convex regions are amenable to representation as an LMI

Centralized Algorithms

- MDS-MAP developed by Shang et al.
- there are n points, suspended in a volume, the positions of the points are not known, but the distance between each pair of points is known
- MDS is an $O(n^3)$ algorithm that uses the Law of Cosines and linear algebra to reconstruct the relative positions of the points based on the pairwise distances
- MDS-MAP performs well on RSSI data alone
- classical metric MDS has fours stages
- Step 1 Gather ranging data from the network, and form a sparse matrix R, where r_{ij} is the range between nodes i and j, or zero if no range was collected (for instance if i and j are physically too far apart)
- Step 2 Run a standard all pairs shortest path algorithm (Dijkstra's, Floyd's) on R to produce a complete matrix of inter-node distances D

Centralized Algorithms: Issues with MDS

- Step 3 Run classical metric MDS on D to find estimated node positions X
- Step 4 Transform the solution X into global coordinates using some number of fixed anchor nodes using a coordinate system registration routine
- MDS-MAP estimates improve as ranging improves, it does not use anchor nodes very well, since it effectively ignores their data until stage 4
- its performance lags behind other algorithms as anchor density increases
- poor asymptotic performance, which is $O(n^3)$ on account of stages 2 and 3
- this problem can be partially ameliorated using coordinate system stitching

Distributed Algorithms

- relevant computation is done on the sensor nodes themselves
- localize nodes directly into the global coordinate space of the beacons
- extrapolate unknown node positions from beacon positions
- four beacon-based distributed algorithms: diffusion, bounding box, gradient multilateration, and APIT
- Diffusion arises from a very simple idea: the most likely position of a node is at the centroid of its neighbors positions
- Diffusion algorithms require only radio connectivity data
- accuracy is poor when node density is low
- Bounding Box algorithm is a computationally simple method of localizing nodes given their ranges to several beacons

Distributed Algorithms

- each node assumes that it lies within the intersection of its beacons' bounding boxes
- the position of a node is then the center of this final bounding box
- accuracy of the bounding box approach is best when nodes' actual positions are closer to the center of their beacons
- bounding box works best when sensor nodes have extreme computational limitations, since other algorithms may simply be infeasible
- otherwise, more mathematically rigorous approaches such as gradient multilateration may be more appropriate

Network-wide localization

- Node localization is determining the location of a single unknown node given a number of nearby references
- A broader problem in sensor systems is that of network localization, where several unknown nodes have to be localized in a network with a few reference nodes
- the network localization problem can rarely be neatly decomposed into a number of separate node localization problems (since there may be unknown nodes that have no reference nodes within range)
- the node localization and ranging techniques described above do often form an integral component of solutions to network localization
- the performance of network localization depends very much on the resources and information available within the network

Network-wide localization

- Several scenarios are possible:
- for instance there may be no reference nodes at all, so that perhaps only relative coordinates can be determined for the unknown nodes
- if present, the number/density of reference nodes may vary (more reference nodes, the lower the network localization error)
- there may be just a single mobile reference
- some network localization approaches are centralized
- the nodes in the network need to be localized only once, post-deployment
- other network localization approaches are distributed, often involving the iterative communication of updated location information

Multi-dimensional scaling

- the node localization problem with defining the network as an undirected graph with vertices V and edges E
- the vertices correspond to the nodes, of which zero or more may be special nodes, called anchors, whose positions are already known
- assume that all the nodes being considered in the positioning problem form a connected graph, i.e., there is a path between every pair of nodes
- classical MDS is the simplest case of MDS: the proximities of objects are treated as distances in a Euclidean space
- the goal of MDS is to find a configuration of points in a multidimensional space such that the inter-point distances are related to the provided proximities by some transformation (e.g., a linear transformation)

Multi-dimensional scaling

- let p_{ij} refer to the proximity measure between objects i and j
- the Euclidean distance between two points $X_i = (x_{i1}; x_{i2}; x_{i3} \dots x_{im})$ and $X_j = (x_{j1}; x_{j2}; x_{j3} \dots x_{jm})$ in an m dimensional space is

$$d_{ij} = \sqrt{\sum_{k=1}^{m} (x_{ik} - x_{jk})^2}$$

- the Euclidean distances are related to the proximities by a transformation $d_{ij} = f(p_{ij})$
- in the classical MDS, a linear transformation model is assumed, $d_{ij} = a + bp_{ij}$
- the distances D are determined so that they are as close to the proximities P as possible

Multi-dimensional scaling

- define $I(P) = D + E$ $I(P)$ is a linear transformation of proximities, E is a matrix of errors
- D is a function of coordinates X, the goal of classical MDS is to calculate X such that the sum of squares of E is minimized, subject to suitable normalization of X
- in classical MDS, P is shifted to the center and coordinates X can be computed from the double centered P through singular value decomposition
- for an $n \times n$ P matrix for n points and m dimensions of each point, it can shown that

$$-\frac{1}{2}\left(p_{ij}^2 - \frac{1}{n}\sum_{i=1}^{n} p_{ij}^2 - \frac{1}{n}\sum_{i=1}^{n} p_{ij}^2 + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} p_{ij}^2\right) = \sum_{k=1}^{m} x_{ik}x_{jk}$$

Multi-dimensional scaling

- the double centered matrix on the left hand side (B) is symmetric
- calculated B and performing singular value decomposition on B gives $B = VAV$
- the coordinate matrix becomes $X = VA^{1/2}$
- retaining the first r largest eigenvalues and eigenvectors ($r < m$) leads to a solution in lower dimension
- this implies that the summation over k runs from 1 to r instead of m
- this is the best low rank approximation in the least-squares sense
- for example, for a 2D network, we take the first 2 largest eigenvalues and eigenvectors to construct the best 2D approximation

Semi-Definite Programming (SDP)

- optimization algorithm that uses techniques of linear programing
- it is a relaxation based method
- trace of a given matrix A, $t_r(A)$ is the sum of the entries on the main diagonal of A
- A symmetrical matrix is called semidefinite if all its eigenvalues are nonnegative and is presented by $A \succeq 0$
- suppose two nodes x_1 and x_2 are within radio range R of each other
- the proximity constraint can be represented as a convex second order cone constraint of the form $\| x_1 - x_2 \| \leq R$ and this can be formulated as a matrix linear inequality

$$\begin{pmatrix} I_2 R & x_1 - x_2 \\ (x_1 - x_2)^T & R \end{pmatrix} \succeq 0$$

Semi-Definite Programming (SDP)

- the mathematical model:
- there are n distinct sensor points in R^{dim} whose locations are to be determined and other m fixed points (called the anchor points) whose locations are known
- known nodes are indicated by a and the unknown nodes by x, so that $X = [x_1, \ldots, x_n, a_1, \ldots, a_m]$
- all $(i, j) \in E$ where $i < j$ and if j is an anchor are denoted by N_a and unknown is denoted by N_x
- the following constraints must be satisfied:

$$\| a_k - x_j \|^2 = d_{kj}^2 \quad \forall (k, j) \in N_a$$

$$\| x_i - x_j \|^2 = d_{ij}^2 \quad \forall (i, j) \in N_x$$

Semi-Definite Programming (SDP)

- when the node distances are known, let $X = [x_1; \ldots; x_n]^T$ be the matrix in $R^{\text{dim} \times (n)}$ that needs to be determined
- define $e_{ij} \in R^n$ with 1 on i th position and with -1 on j th position and everywhere else zeros; I_{dim} is the identity matrix, the constraints can be written:

$$(a_k; e_j)^T [I_{\text{dim}}; X^T][I_{\text{dim}}; X](a_k; e_j) = d_{kj}^2 \; \forall (k,j) \in N_a$$

$$e_{ij}^T X^T X e_{ij} = d_{ij}^2 \; \forall (i,j) \in N_x$$

- need to find a symmetric matrix $Y \in R^{\text{dim} \times \text{dim}}$ and X that satisfy the following constraints:

$$(a_k; e_j)^T \begin{pmatrix} I_{\text{dim}} & X \\ X^T & Y \end{pmatrix} (a_k; e_j) = d_{kj}^2 \; \forall (k,j) \in N_a$$

Semi-Definite Programming (SDP)

$$e_{ij}^T Y e_{ij} = d_{ij}^2 \;\; \forall (i,j) \in N_x$$
$$Y = X^T X$$

- this is the SDP formulation of the problem of WSN localization
- the constraint

$$Y = X^T X$$

  is relaxed with $Y \succeq X^T X$
- this condition can be written as

$$Z = \begin{pmatrix} I_{\text{dim}} & X \\ X^T & Y \end{pmatrix} \succeq 0$$

Semi-Definite Programming (SDP)

- m known points (anchors) $a_k \in \mathcal{R}^2, k = 1, \ldots, m$ and n unknown points (sensors) $x_j \in \mathcal{R}^2, j = 1, \ldots, n$
- for a pair of two points in N_e, Euclidean distance measurement $\hat{d}_{kj}$ between a_k and x_j or $\hat{d}_{ij}$ between x_i and x_j
- for a pair of two points in N_l, a distance lower bound $\underline{r}_{kj}$ between a_k and x_j or $\underline{r}_{ij}$ between x_i and x_j
- for a pair of two points in N_u, a distance upper bound $\bar{r}_{kj}$ between a_k and x_j or $\bar{r}_{ij}$ between x_i and x_j
- the localization problem is to find x_js such that

$$\begin{aligned}
||x_i - x_j||^2 = (\hat{d}_{ij})^2, ||a_k - x_j||^2 = (\hat{d}_{kj})^2, \forall (i,j), (k,j) \in N_e \\
||x_i - x_j||^2 \geq (\underline{r}_{ij})^2, ||a_k - x_j||^2 \geq (\underline{r}_{kj})^2, \forall (i,j), (k,j) \in N_l \\
||x_i - x_j||^2 \leq (\bar{r}_{ij})^2, ||a_k - x_j||^2 \leq (\bar{r}_{kj})^2, \forall (i,j), (k,j) \in N_u
\end{aligned}$$

Semi-Definite Programming (SDP)

- due to measurement error x_js can be chosen to minimize the sum of errors:

$$\begin{aligned}
\min \sum_{i,j \in N_e, i<j} ||| x_i - x_j \|^2 - (\hat{d}_{ij})^2 | + \sum_{k,j \in N_e} ||| a_k - x_j \|^2 - (\hat{d}_{kj})^2 | \\
+ \sum_{i,j \in N_l, i<j} \left(\| x_i - x_j \|^2 - (\underline{r}_{ij})^2 \right)_- + \sum_{k,j \in N_l} \left(\| a_k - x_j \|^2 - (\underline{r}_{kj})^2 \right)_- \\
+ \sum_{i,j \in N_u, i<j} \left(\| x_i - x_j \|^2 - (\bar{r}_{ij})^2 \right)_+ + \sum_{k,j \in N_u} \left(\| a_k - x_j \|^2 - (\bar{r}_{kj})^2 \right)_+
\end{aligned}$$

- $(u)_-$ and $(u)_+$ defined as $(u)_- = \max\{0, -u\}$ and $(u)_+ = \max\{0, u\}$
- by introducing slack variables and relaxing $Y = X^T X$ to $Y \succeq X^T X$, the problem can be rewritten as a standard SDP problem