Reconstruction of Euclidean Embeddings in Dense Networks

Sarah Costrell[1], Subhrajit Bhattacharya[2], and Robert Ghrist[1,2]

(1)Department of Electrical and Systems Engineering, University of Pennsylvania
(2)Department of Mathematics, University of Pennsylvania

Overview

The problem: determining the Euclidean embedding of a dense, planar sensor network.
Assumptions: each sensor has a binary protocol to detect neighboring sensors within a fixed radius, sensors densely distributed
Contributions: an algorithm to identify landmark nodes in the network whose Euclidean embedding is "close" to the vertices of an ideal hexagonal lattice, theoretical bounds on the error between the reconstructed lattice embedding and its ground truth embedding
Applications: GPS-free localization, mapping, environmental monitoring

Pivot Function Definition

Given $v_1, v_2, v_3 \in V$ such that $d_G(v_1, v_2) = d_G(v_1, v_3) = d_G(v_2, v_3) = N$, define $P : V^3 \to V$ such that $P(v_1, v_2; v_3) = v \in V : d_G(v, v_1) = d_G(v, v_2) = N, \lfloor N\sqrt{3} \rfloor \leq d_G(v, v_3) \leq \lceil N\sqrt{3} \rceil$. We say v_1 and v_2 are the parent points and v_3 is the pivot point with respect to v. Note that there could be many feasible $v \in V$, in which case we select an arbitrary one, and that denseness guarantees the existence of at least one such v.

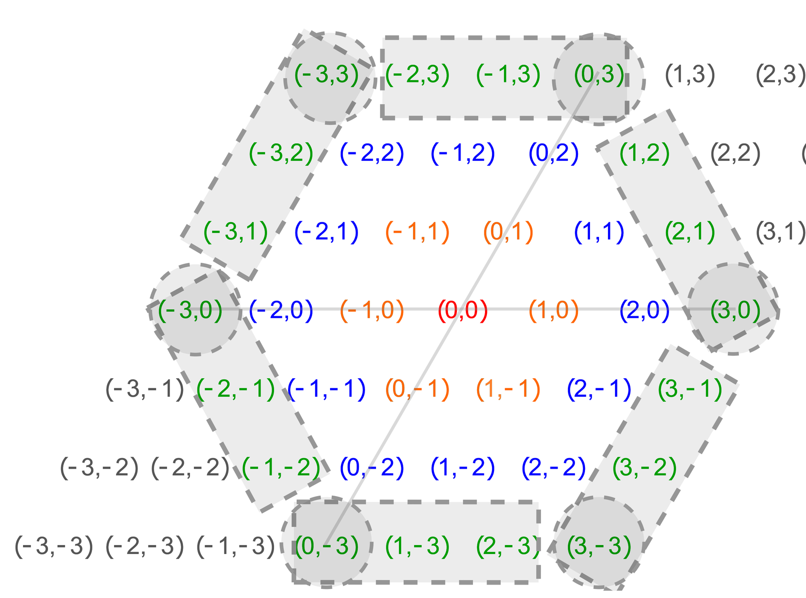

Figure 1: The triangular coordinate system for $-3 \leq x \leq 3$, $-3 \leq y \leq 3$ with several lattice level level-sets shown in different colors. On lattice level 3, sides are denoted by rectangular boxes and corners by discs.

Analytical Results

- Lemma: Let the network G be ϵ-dense, i.e., for every $x \in \mathbb{R}^2$, there is at least one vertex $w \in V$ such that $r(w) \in B_\epsilon(x)$. Then, the N-hop metric distance is bounded with respect to N and R: If $d_G(u, v) = N$, then $(R - 2\epsilon)(N - 1) \leq d(u, v) \leq RN$.
- Proposition: Let $\Delta l := \frac{R}{2} + \epsilon(N - 1)$. Let e_n be the accumulated error at a point requiring n parent points to exist before it can be created. Then for large n, $e_n \leq \Delta l(1.81)^n$.

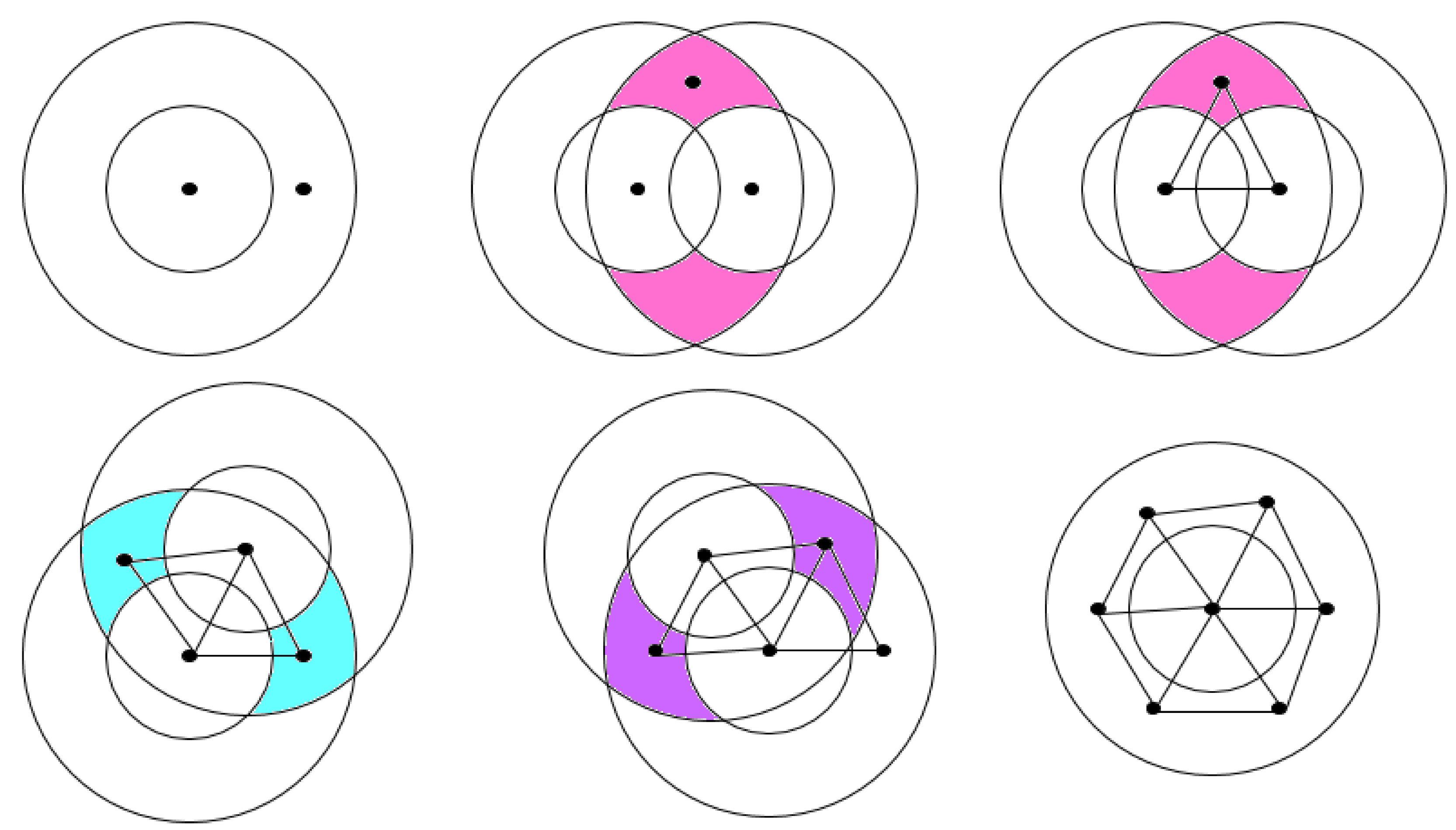

Figure 2: Setting up the initial hexagon using the pivot function.

Problem Statement

We compute an approximately distance-preserving embedding of a dense sensor network into $\mathbb{R}^2$. In doing so, we consider the undirected, unweighted connectivity graph $G = (V, E)$, where V is the set of nodes corresponding to the set of sensors and E is the set of all pairs of sensors (v, w) such that the Euclidean embeddings of v and w are within a fixed distance R of one another. Our algorithm takes as its inputs a graph G and a desired highest lattice level H (see Fig. 1) and outputs a subset of size $1+3H(H+1)$ of the vertex set V, specifically, vertices v_i such that each its embedding is as close as possible to the corresponding vertex of an ideal hexagonal lattice.

High-level Algorithm Description

1. Choose a random initial sensor as a landmark node and find its N-hop neighborhood.
2. Choose a second sensor (landmark node) from the N-hop neighborhood of the first sensor.
3. Take the intersection of the N-hop neighborhoods of the first two sensors and choose the third sensor from this intersection.
4. Use the pivot function to generate the next landmark node from the previous three (see Figure 2).
5. Continue in a 'circular' manner, using previous landmark nodes to fill out successive lattice levels.

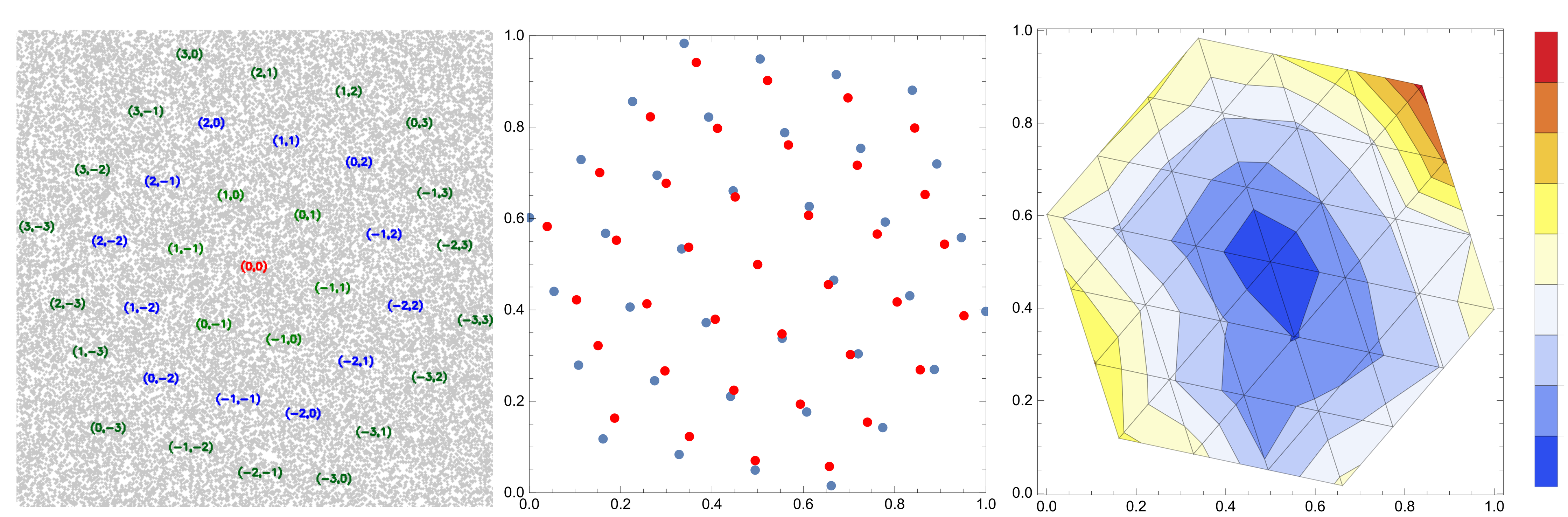

Figure 3: Results of numerical simulation of algorithm on 34,000 vertices with $N = 9$, $R = 0.02$, lattice level 3. **Left:** Fig. generated using OpenCV with algorithmic output for levels 0 through 3. **Center:** Points generated by algorithm are plotted in red, corresponding points ideal lattice points plotted in blue. **Right:** Contour plot of observed error in Euclidean distance between the algorithmic output and the ideal lattice points.

Algorithm Pseudocode

Construction of sides for lattice level > 1; Alg refers to points created by algorithm, indexed as seen in Figure 1.

$highestLevel \leftarrow H$ ▷ H is the desired highest level
$C \leftarrow 1$ ▷ C is the level currently being built upon
for $C < highestLevel$, step 1 **do**
▷ The following loop builds the side $c > 0$, $x + y = c$:
$\bar{x} \leftarrow 0$, $\bar{y} \leftarrow C$ ▷ $\bar{x}$ and $\bar{y}$ are temporary variables
for $i := 0$ to $i < C$, step 1 **do**
$\mathrm{Alg}(\bar{x}+1, \bar{y}) \leftarrow P(\mathrm{Alg}(\bar{x}, \bar{y}), \mathrm{Alg}(\bar{x}+1, \bar{y}-1); \mathrm{Alg}(\bar{x}, \bar{y}-1))$
$\bar{x} \leftarrow \bar{x}+1$, $\bar{y} \leftarrow \bar{y}-1$
$\mathrm{Alg}(C+1, 0) \leftarrow P(\mathrm{Alg}(C, 0), \mathrm{Alg}(C, 1); \mathrm{Alg}(C-1, 1))$
▷ Previous line creates corner point
▷ The following loop builds the side $c > 0$, $x = c$:
$\bar{x} \leftarrow C$, $\bar{y} \leftarrow 0$
for $i := 0$ to $i < C$, step 1 **do**
$\mathrm{Alg}(\bar{x}+1, \bar{y}-1) \leftarrow P(\mathrm{Alg}(\bar{x}, \bar{y}), \mathrm{Alg}(\bar{x}, \bar{y}-1); \mathrm{Alg}(\bar{x}-1, \bar{y}))$
$\bar{y} \leftarrow \bar{y}-1$
$\mathrm{Alg}(C+1, -C-1) \leftarrow$
$P(\mathrm{Alg}(C, -C), \mathrm{Alg}(C+1, -C); \mathrm{Alg}(C, -C+1))$
▷ The following loop builds the side $c < 0$, $y = c$:
$\bar{x} \leftarrow C$, $\bar{y} \leftarrow -C$
for $i := 0$ to $i < C$, step 1 **do**
$\mathrm{Alg}(\bar{x}, \bar{y}-1) \leftarrow P(\mathrm{Alg}(\bar{x}, \bar{y}), \mathrm{Alg}(\bar{x}-1, \bar{y}); \mathrm{Alg}(\bar{x}-1, \bar{y}+1))$
$\bar{x} \leftarrow \bar{x}-1$
$\mathrm{Alg}(0, -C-1) \leftarrow P(\mathrm{Alg}(0, -C), \mathrm{Alg}(1, -C-1); \mathrm{Alg}(1, -C))$
▷ The following loop builds the side $c < 0$, $x + y = c$:
$\bar{x} \leftarrow 0$, $\bar{y} \leftarrow -C$
for $i := 0$ to $i < C$, step 1 **do**
$\mathrm{Alg}(\bar{x}-1, \bar{y}) \leftarrow P(\mathrm{Alg}(\bar{x}, \bar{y}), \mathrm{Alg}(\bar{x}-1, \bar{y}+1); \mathrm{Alg}(\bar{x}, \bar{y}+1))$
$\bar{x} \leftarrow \bar{x}-1$, $\bar{y} \leftarrow \bar{y}+1$
$\mathrm{Alg}(-C-1, 0) \leftarrow$
$P(\mathrm{Alg}(-C, 0), \mathrm{Alg}(-C, -1); \mathrm{Alg}(-C+1, -1))$
▷ The following loop builds the side $c < 0$, $x = c$:
$\bar{x} \leftarrow -C$, $\bar{y} \leftarrow 0$
for $i := 0$ to $i < C$, step 1 **do**
$\mathrm{Alg}(\bar{x}-1, \bar{y}+1) \leftarrow P(\mathrm{Alg}(\bar{x}, \bar{y}), \mathrm{Alg}(\bar{x}, \bar{y}+1); \mathrm{Alg}(\bar{x}+1, \bar{y}))$
$\bar{y} \leftarrow \bar{y}+1$
$\mathrm{Alg}(-C-1, C+1) \leftarrow$
$P(\mathrm{Alg}(-C, C), \mathrm{Alg}(-C-1, C); \mathrm{Alg}(-C, C-1))$
▷ The following loop builds the side $c > 0$, $y = c$:
$\bar{x} \leftarrow -C$, $\bar{y} \leftarrow C$
for $i := 0$ to $i < C$, step 1 **do**
$\mathrm{Alg}(\bar{x}, \bar{y}+1) \leftarrow P(\mathrm{Alg}(\bar{x}, \bar{y}), \mathrm{Alg}(\bar{x}+1, \bar{y}); \mathrm{Alg}(\bar{x}+1, \bar{y}-1))$
$\bar{x} \leftarrow \bar{x}+1$
$\mathrm{Alg}(0, C+1) \leftarrow P(\mathrm{Alg}(-1, C+1), \mathrm{Alg}(0, C); \mathrm{Alg}(-1, C))$

Acknowledgements

Authors supported by US DoD contracts FA9550-12-1-0416 and N00014-16-1-2010.

Contact Information

- Author 1: costrell@seas.upenn.edu
- Author 2: subhrabh@math.upenn.edu
- Author 3: ghrist@math.upenn.edu