Node Localization in an Obstacle-Rich Environment

Nick Kantack
5-3-2019

Abstract

I present a node localization algorithm ("Break Formation," or BF) specifically designed to localize wireless sensors from a received signal strength indicator (RSSI) in the presence of significant environmental interference with node-to-node line of sight distance measurements. This algorithm accomplishes two goals. The first is to identify the environmental noise in node-to-node distance measurement to allow accurate node-to-node distance calculation in the presence of obstacles. The second is to extract the environmental noise in a way that allows basic mapping of the obstacles. I demonstrate through several simulations that BF outperforms several wall-sensitive localization algorithms in a highly obstructive environment and provides a method of constructing a map of the environment as a byproduct of its node localization.

Introduction

Node localization is a spatial tracking problem with a wide variety of useful applications. In particular, node localization can provide a strong strategic advantage to personnel teams navigating an environment (e.g. SWAT, soldiers, or firefighters entering a building). Geolocation systems like GPS have substantial power requirements, significant latency, and close-quarters accuracy limitations which prevent such systems from being practical localization solutions for personnel teams on the basis of cost, refresh rate, error, and access to GPS signal in restrictive environments. Therefore, there remains a need for localization technology which provides a low cost, fast, and accurate means of tracking personnel team members in uncharted, obstructive, and evolving environments.

This paper provides a method for node localization which is part of a broader initiative to develop a proof of concept team tracking system for first responders. In particular, the goal of this initiative is to develop lightweight body-worn sensors which collectively compute the map of the team and provide each team member a visual projection of the location of his or her teammates through an augmented reality interface.

Node localization in the absence of obstacles is well studied and many effective algorithms exist for generating accurate node maps from measured node-to-node distances [1][2]. The foundational method for generating such a map is through a method called Classical Multidimensional Scaling, or Classical MDS. Classical MDS is a dimensional reduction method that can generate a mapping (in a real or abstract space) of an arbitrary number of nodes using only the set of node-to-node distances or dissimilarities. Therefore, Classical MDS is clearly applicable to the task of generating a mapping in n dimensional space from known node-to-node distances. However, Classical MDS presupposes accurate node-to-node distances in order to obtain an accurate localization map.

To overcome the sensitivity of Classical MDS to erroneous node-to-node distance estimations, many modified version of Classical MDS have emerged. In many cases, an obstacle which breaks the line of sight between two nodes will cause the two nodes to be out of measurement range with one another. In this case, several algorithms exist for populating the missing elements of the node-to-node distance matrix, many containing heuristics to prevent exaggeration of distances which were not directly measured[3][5]. These algorithms make use of the fact that the influence of the environment is obvious from the terms missing in the distance matrix. These methods do not accommodate the situation in which obstacles cause "contaminated" distance measurements to enter the distance matrix.

Some methods exist for correcting the node-to-node distance matrix when non-line-of-sight (NLOS) communication makes a significant contribution to the RSSI (e.g. nodes in a large room with small obstacles) in addition to line-of-sight (LOS) RSSI. Li et. al. [1] propose fitting a Gaussian Mixed Model to the measured RSSI and identifying a corrected RSSI-based distance matrix for MDS optimization. In relatively open indoor environments, this method can improve the performance of MDS substantially. However, this method is limited in its ability to reconstruct an accurate distance matrix when obstructed LOS RSSI is the dominant signal, as in the case of nodes which are each in separate, closed rooms. Furthermore, indoor MDS-based algorithms typically seek to cancel out the influence of walls and obstacles on the distance matrix without an explicit method for cleanly separating the influence in a way that would facilitate a reconstruction of a map of the environment.

This paper proposes an algorithm design which demonstrated in simulation the ability to cleanly separate the influence of obstacles on RSSI and the true distances in order to generate accurate maps of the nodes and accurate maps of the environment.

Problem Formulation

Let us construct a 3 way, $N \times N \times T$ tensor $\mathcal{S}$ which is the dissimilarity tensor. N is the number of nodes to be localized. T is the number of discrete instants in time for which node-to-node distance measurements are taken. Let $S(i,j,t)$ be the distance between the ith and jth nodes at time t as estimated from the RSSI between the two nodes.

$\mathcal{S}(:,:,t)$ is a $N \times N$ matrix which represents the measured node-to-node distances at time t. It is possible to perform Classical MDS upon this $\mathcal{S}(:,:,t)$ matrix to obtain a mapping of the nodes. To perform Classical MDS, we first compute a double centered, squared distance matrix B.

$$B = -\frac{1}{2} J \left(\mathcal{S}(:,:,t) * \mathcal{S}(:,:,t) \right) J \tag{1}$$

where J is the centering matrix

$$J = \left(\frac{N-1}{N} \right) I^{N \times N} - \frac{1}{N} 1^{N \times N} \tag{2}$$

where $1^{N \times N}$ is an $N \times N$ matrix with each element equal to 1. $I^{N \times N}$ is the $N \times N$ identity matrix. The node localization problem is reduced to

$$\min_X ||B - XX^T||_F^2 \tag{3}$$

where X is a $N \times n$ matrix. n is the number of dimensions of the localization space. $X(i,j)$ is the value of the jth coordinate of the ith node. Under these conditions, the optimal X is given by the eigendecomposition of B. If $U \Lambda V^T$ is the SVD of B,

$$X = U(:,n) \Lambda .^{\frac{1}{2}} \tag{4}$$

where $\Lambda .^{\frac{1}{2}}$ denotes the element-wise square root of Λ.

Let us represent the solution to the long term node localization problem as computing the $N \times n \times T$ map tensor $\mathcal{X}$.

One possible method for determining $\mathcal{X}$ is to let $\mathcal{X}(:,:,t)$ be the classical MDS solution for the dissimilarity matrix $\mathcal{S}(:,:,t)$.

Derivation of the "Break Formation" Method

Because of the influence of a wall on RSSI, the dissimilarity matrices $\tilde{\mathcal{S}}(:,:,t)$ contains some square distances which are grossly exaggerated, specifically for nodes separated by the wall. Apply classical MDS to the unaltered, measured $\tilde{\mathcal{S}}(:,:,t)$ produces a significantly distorted map (see **Figure 1**).

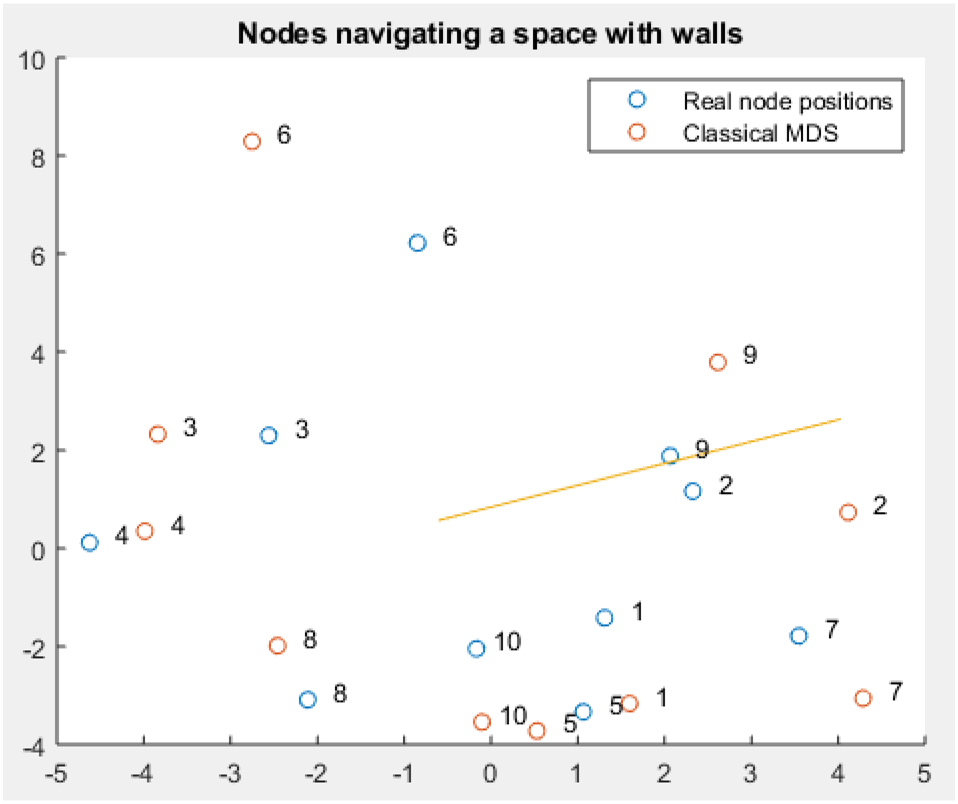


Figure 1.
Description

The "Break Formation Localization" algorithm (BFL) utilizes classical MDS along with a method for countering the distortion from walls and obstacles. Let us make two assumptions to facilitate algorithm design.

Let us make two assumptions to facilitate algorithm design.

- If an obstacle breaks the line of sight between two nodes, it applies a known scaling to the measured distance between these two nodes
- At $t = 0$, no obstacles are breaking the line of sight between any two nodes (e.g. the team starts from a huddle).

From the first assumption, we can adopt a more detailed model for the node-to-node distances. Let

$$\tilde{\mathcal{S}} = (1^{N\times N\times T} + q\mathcal{Q}) * \mathcal{S} \tag{5}$$

$\tilde{\mathcal{S}}$ is the measured dissimilarity tensor, and $\mathcal{S}$ is the dissimilarity matrix with the obstacle influence removed (but there may still be measurement noise in $\mathcal{S}$). $\mathcal{Q}$ is a $N \times N \times T$ tensor of 1's and 0's where

$$\mathcal{Q}(i,j,t) = \begin{Bmatrix} 1 & \text{if an obstacle obstructs the line of sight between node } i \text{ and } j \\ 0 & \text{otherwise} \end{Bmatrix} \tag{6}$$

$q + 1$ is the known scaling that is applied to measured node-to-node distances when influenced by an obstacle (we will later generalize for an unknown scaling).

With this formulation, it is possible to obtain an estimate for $\mathcal{S}$ from $\tilde{\mathcal{S}}$ as long as an accurate estimation of $\mathcal{Q}$ is possible. Note that at the moment a wall breaks the line of sight of two nodes (e.g. $t = t^*$) the dissimilarity tensor $\tilde{\mathcal{S}}$ exhibits an abrupt change.

$$\frac{d\tilde{\mathcal{S}}}{dt}|_{t=t^*} = \frac{\tilde{\mathcal{S}}(i,j,t^*) - \tilde{\mathcal{S}}(i,j,t^*-1)}{\tau} \text{ which is usually large} \tag{7}$$

In many cases (when the nodes are sufficiently spaced and the refresh rate of the system is high), $\frac{d\tilde{\mathcal{S}}}{dt}|_{t=t^*}$ may exceed the maximum distance a node could have traversed, $v_{max}\tau$. This provides a useful criterion for generating an accurate estimate for $\mathcal{Q}$. For instance, let

$$\mathcal{Q}(i,j,t) = u\left(|\tilde{\mathcal{S}}(i,j,t) - \tilde{\mathcal{S}}(i,j,t-1)| - v_{max}\tau\right) \tag{8}$$

where $u(t)$ is the Heaviside step function. To recite this criterion, let a wall's influence be inferred for any change in node-to-node distance which could not be generated from pure node motion. This criterion is conservative in its inference of walls, which helps prevent problem obfuscation from inferring an excessive number of obstacles.

For RSSI based distance measurements, noise can be considerable in size, often as large as 20%. Therefore, the wall inference criterion above can be fragile in the presence of noise and is easily triggered when measurement noise flips from time step to time step causing a perceived jump in the distance matrix attributed to a wall. To prevent misfiring of the wall inference criterion, one can adopt a low-pass filtered criterion for $\mathcal{Q}$.

$$\boxed{\mathcal{Q}(i,j,t) = u\left(|\tilde{\mathcal{S}}(i,j,t) - \tilde{\mathcal{S}}(i,j,t-m)| - m v_{max}\tau\right)} \tag{9}$$

where m is an integer which delays the advent of a wall inference in order to significantly reduce the probability of an incorrect wall inference caused by noise. Details on an optimal choice of m appear in the Results section.

A cost function for further optimization

It is important to note that classical MDS does not yield solutions for the tensor slabs $\mathcal{X}(:,:,t)$ which are linear least squares optimal for the true distance tensor slabs $\mathcal{S}(:,:,t)$ since the matrix $\mathcal{S}$ was modified (through a double centering and squaring) to generate the tensors slabs $\mathcal{B}(:,:,t)$ which were used in the classical MDS computation. Therefore, a gradient descent method can further improve the quality of the node location estimation $\mathcal{X}$.

For gradient descent for node localization, the following stress function is typically used.

$$\text{stress} = 2\sum_{t=1}^{T}\sum_{i=1}^{N}\sum_{j=i+1}^{N} w_{ij}(\mathcal{S}(i,j,t) - \mathcal{S}_{\mathcal{X}}(i,j,t))^2 \tag{10}$$

$\mathcal{S}_{\mathcal{X}}(i,j,t)$ is the distance between nodes i and j at time t as calculated from the estimated $\mathcal{X}(:,:,t)$. w_{ij} is a weight factor which is zero when nodes i and j are out of range of one another. For preliminary simplicity and to contextualize the problem to close quarters indoors, we will assume $w_{ij} = 1$ for all combinations i, j. Therefore, we can write

$$\text{stress} = \sum_{t=1}^{T} ||\mathcal{S}(:,:,t) - \mathcal{S}_{\mathcal{X}}(:,:,t)||_F^2 \tag{11}$$

By taking a derivative the following gradient can be found.

$$\frac{\partial \text{stress}}{\partial \mathcal{X}(i,q,t)} = \sum_{j=1}^{N}(1-\delta_{ij})\frac{(\mathcal{S}(i,j,t) - \mathcal{S}_{\mathcal{X}}(i,j,t))(\mathcal{X}(i,q,t) - \mathcal{X}(j,q,t))}{\mathcal{S}_{\mathcal{X}}(i,j,t)} \tag{12}$$

where δ_{ij} is the Kronecker delta function. In practice, it is beneficial to add a smoothness constraint to the cost function specifically promoting smoothness of $\mathcal{X}$ along the temporal mode. Therefore, BF makes use of the following modified stress function.

$$\text{stress} = \sum_{t=1}^{T} ||\mathcal{S}(:,:,t) - \mathcal{S}_{\mathcal{X}}(:,:,t)||_F^2 + \mu\sum_{t=1}^{T} ||\mathcal{X}(:,:,t) - \mathcal{X}(:,:,t-1)||_F^2 \tag{13}$$

This stress function yields the following gradient

$$\boxed{\frac{\partial \text{stress}}{\partial \mathcal{X}(i,q,t)} = \sum_{j=1}^{N}(1-\delta_{ij})\frac{(\mathcal{S}(i,j,t) - \mathcal{S}_{\mathcal{X}}(i,j,t))(\mathcal{X}(i,q,t) - \mathcal{X}(j,q,t))}{\mathcal{S}_{\mathcal{X}}(i,j,t)} + 4\mu(\mathcal{X}(i,q,t) - \mathcal{X}(i,q,t-1))} \tag{14}$$

This gradient was used for the gradient descent component of the BF algorithm, which is outlined in the following section.

Algorithm Design

The BF algorithm is illustrated below.

Break Formation Algorithm I (performed after data collected)

```
$\tilde{\mathcal{S}} \leftarrow$ Measured node-to-node RSSI (interpreted as distance)
$\mathcal{X}(:,:,1) \leftarrow$ Solution to classical MDS on $\tilde{\mathcal{S}}(:,:,1)$ (no walls)
% Calculate $\mathcal{Q}$
for $t = m : T$
    for $i = 1 : N$
        for $j = i + 1 : N$
            $\mathcal{Q}(i,j,t) \leftarrow \mathcal{Q}(i,j,t) + u(|\tilde{\mathcal{S}}(i,j,t) - \tilde{\mathcal{S}}(i,j,t-m)|/\tau - mv_{max}\tau)$    % wall detected
            $\mathcal{Q}(i,j,t) \leftarrow \mathcal{Q}(i,j,t) - u(-|\tilde{\mathcal{S}}(i,j,t) - \tilde{\mathcal{S}}(i,j,t-m)|/\tau - mv_{max}\tau)$    % wall passed
        end
    end
end
% Extract the walls' influence from $\tilde{\mathcal{S}}$
for $t = 2 : T$
    $\tilde{\mathcal{S}}(:,:,t) \leftarrow \tilde{\mathcal{S}}(:,:,t)./(1 + q\mathcal{Q}(:,:,t))$    % (./ is element-wise division)
end
for $t = 2 : T$
    $\mathcal{X}(:,:,t) \leftarrow$ MDS Solution for $\tilde{\mathcal{S}}(:,:,t)$
end
% Gradient descent to improve solution
oldStress $\leftarrow$ stress($\mathcal{X}$)
$\tilde{\mathcal{X}} \leftarrow \mathcal{X}$
change = -1
while (change ¡ 0)
    $\mathcal{X} \leftarrow \tilde{\mathcal{X}}$
    for $t = 1 : T$
        for $i = 1 : N$
            for $q = 1 : n$
                $\mathcal{X}(i,q,t) \leftarrow \mathcal{X}(i,j,t) - \alpha\partial(\text{stress})/\partial\mathcal{X}(i,q,t)$
            end
        end
    end
    newStress = stress($\mathcal{X}$)
end % Perform rotation and flip as needed to align with anchors.
```

Results

Simulation Design

I wrote a Matlab code to simulate a team of nodes navigating a space with walls. The simulation construction was as follows.

Node Localization Simulation Design

Allocate memory for an $10 \times 2 \times 100$ real-node-position tensor `Xr` (10 nodes in 2 dimension for 100 moments in time).
Assign random starting locations in $0 < x < 10,\ 0 < y < 10$ for the nodes (`Xr(:,:,1)`).
Allow nodes to randomly walk as time evolves (`Xr(i,j,t) - Xr(i,j,t-1) + 0.2*rand() + 0.1` with a drift).
Compute $10 \times 10 \times 100$ real distance tensor `Sr`.
Add random noise of 15% to measured distances (`S(i,j,t) = Sr(i,j,t)*(1+normrnd(0, 0.15))`).
Apply wall effect (`S(i,j,t) = Sr(i,j,t)*q` if wall breaks line of sight between nodes i and j).

This simulation generates a realistic dissimilarity matrix that might arise from node-to-node distance measurements as the nodes navigate a wall containing environment. Specifically, random measurement noise and systematic wall-effect error are taken into account. I applied various algorithms to extract accurate estimates for $\mathcal{X}_r$ from the generated $\tilde{\mathcal{S}}$ and their performances are compared later in this section. As a side note, the simulation did not include a rigorous method for aligning calculated mappings relative to fixed anchor points. Such a process is well studied and elegant implementations have been developed [4][6]. Rather, for comparison purposes, each algorithm in the analysis below was rotated and flipped as needed to reach a least squares orientation match with the real node locations. Doing so is "cheating" in the sense that no algorithm would have the *a priori* information needed to make this alteration. However, each algorithm was allowed to cheat in the same way so that this study could specifically focus on comparing each algorithm's ability to reproduce the real distance matrix.

To compare the performance of different algorithms, I devised the following error parameter.

$$\text{Error Proxy} = \frac{||\mathcal{X}(:,:,t) - \mathcal{X}_r(:,:,t)||_F}{||\mathcal{S}(:,:,t)||_F} \tag{15}$$

where $\mathcal{X}_r$ is the tensor of the real locations of the nodes. The Error Proxy can be calculated for a particular moment in time t, or it can be averaged over all time for a more general comparison of solutions.

Algorithm Performance

Under various conditions I compared the performance of four algorithms: Classical MDS, Classical MSD with gradient descent (MDS-GD), Break Formation (BF), and Break Formation with gradient descent (BF-GD). For each simulation, it is insightful to example the error in each method plotted in time. This allows a clear comparison of how well each algorithm handled the onset of walls in the environment.

In the simulation, the nodes begin with some initial spatial distribution and each node proceeds with a random walk that drifts towards a wall. The group move about the wall and past it. A plot of the trajectories of the 10 points about a wall is shown below for illustration.

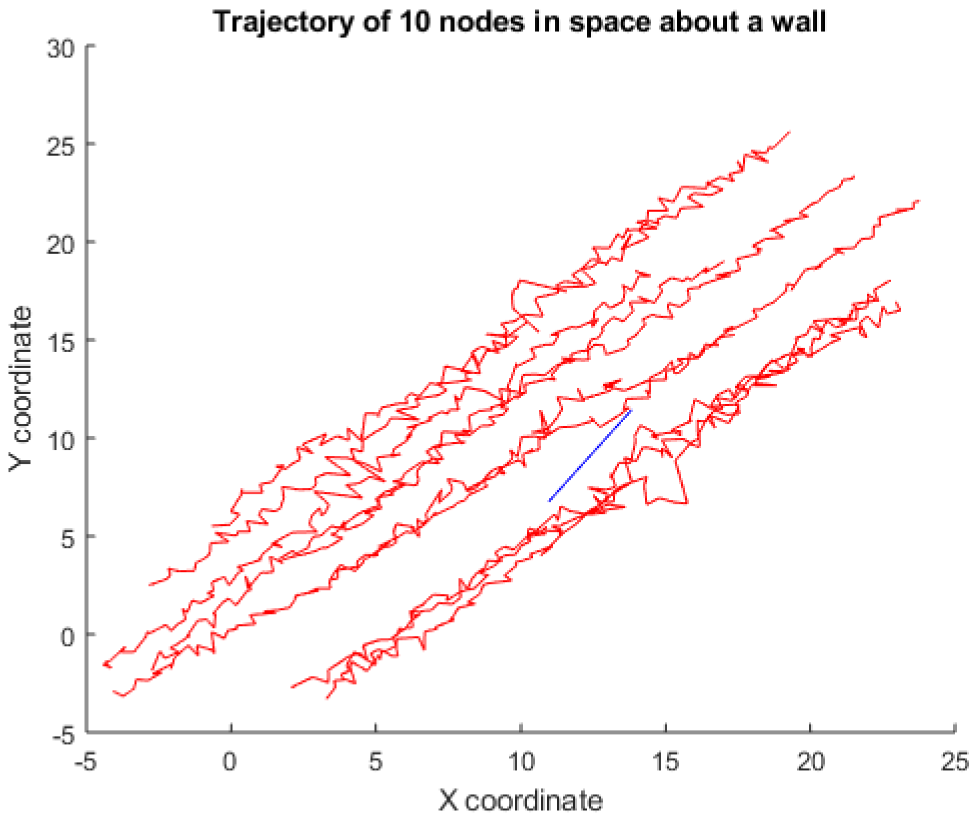


Figure X.
The trajectories (in red) of 10 points as they navigate an environment (starting in lower left and moving towards upper right). A wall (blue line) is roughly in the center of the plotted space.

For this particular simulation, the performance of the four algorithms were as shown below.

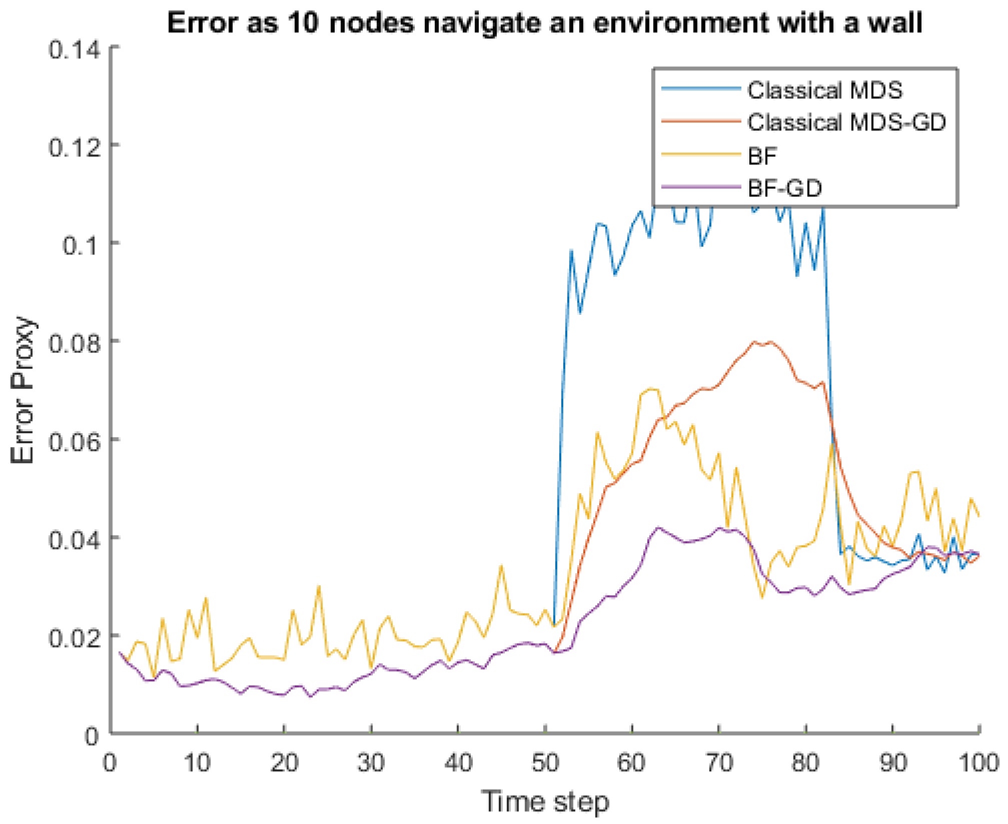


Figure X.
The evolution of the error proxy for each algorithm for this particular run of the simulation.

BF and Classical MDS are superimposed before the advent of the wall's influence. Once the wall begins to separate nodes, there is a divergence with BF better handling the wall's influence by actively seeking to separate it from the measured node-to-node distances. The application of additional gradient descent optimization improves both Classical MDS and BF.

Below is shown a snapshot in time during the simulation. To keep the figure clear, both gradient descent algorithms were omitted from the plot.

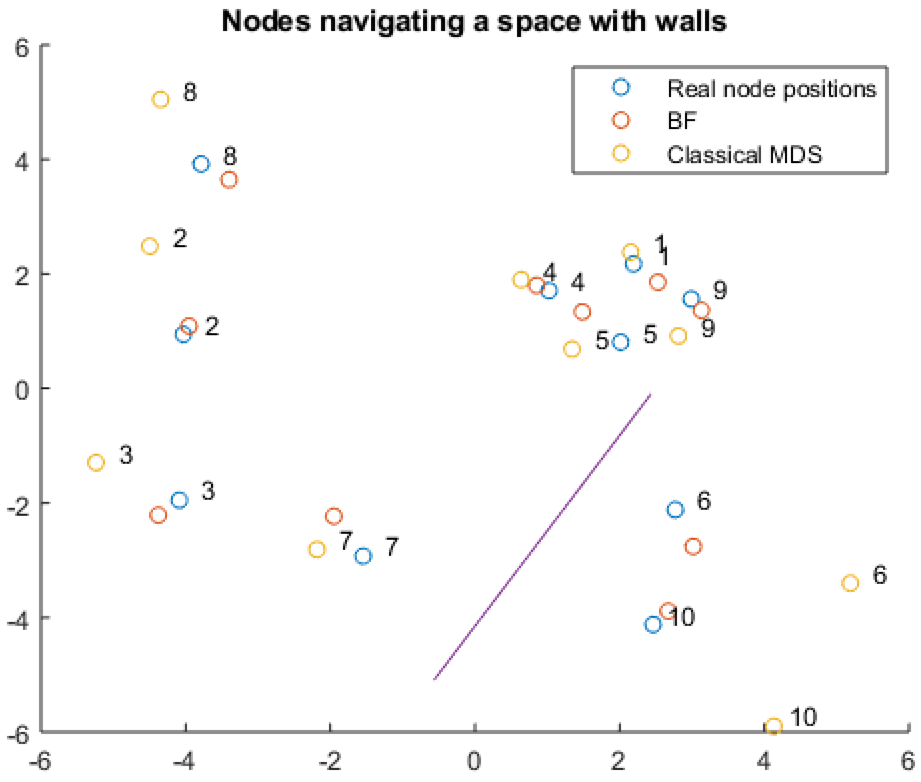


Figure X.
A snapshot in time of the simulation comparing classical MDS to BF.

To provide a clearer picture of the typical results of the simulation, three more runs of the simulation are shown in the next figure.

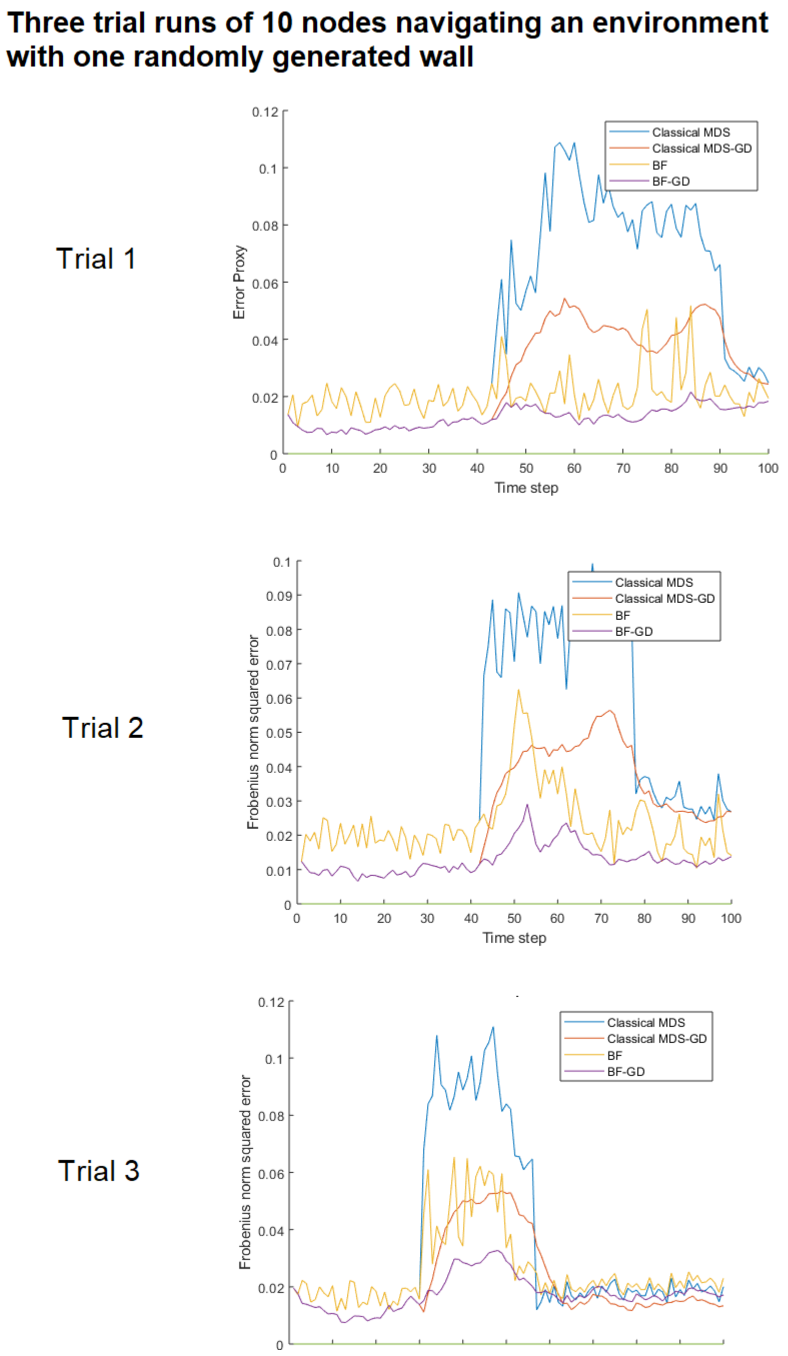


Figure X.

A comparison of three different trials of the simulation (each with a unique initialization).

To conclude the case study with a single wall, a run of 20 trials was conducted and the error proxies for each method were normalized with respect to the error proxy of Classical MDS.

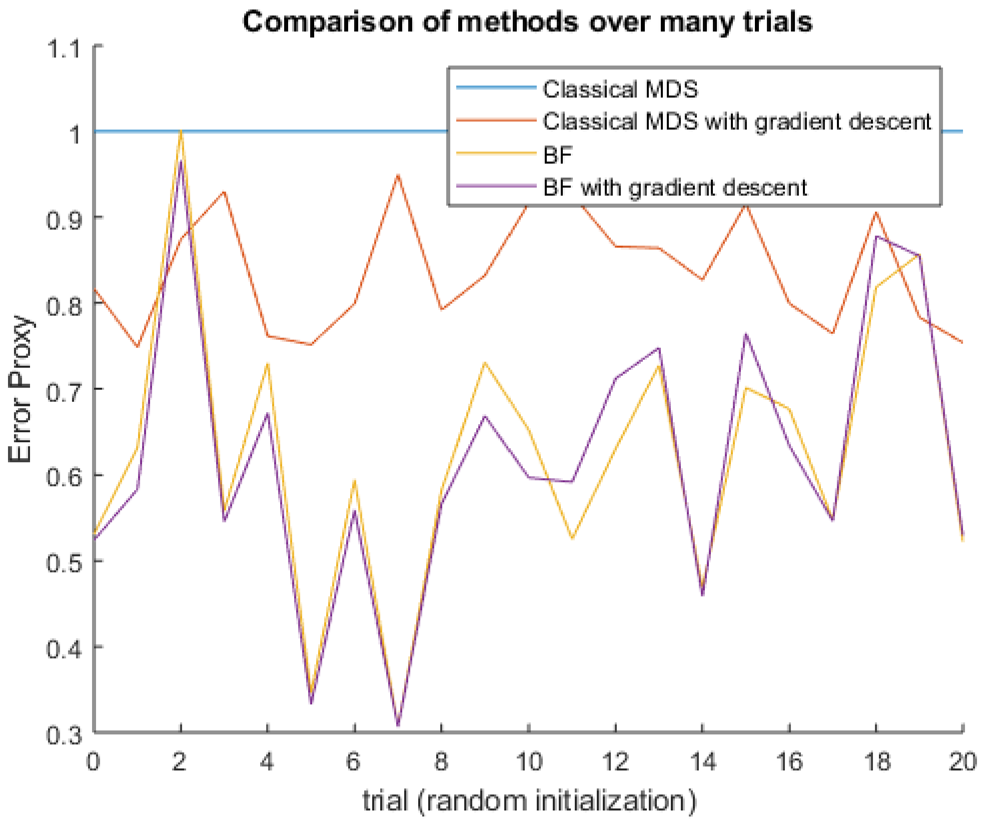


Figure X.
In the simple case of one wall and 15% random noise on estimated node-to-node distances, BF demonstrated considerably higher accuracy node position estimations with an average error reduction of about 30% relative to classical MDS.

Optimizing Algorithm Parameters

Recall that the criterion for detecting a wall and incrementing an element of $\mathcal{Q}$ included a delay factor m. The effect of m is to reduce the likelihood that random measurement noise would trigger the criterion and cause a false positive wall detection. In practice, $m = 1$ was indeed prone to very inaccurate wall estimations. I conducted a study which examined the relative performance of BF over classical MDS for a range of values of m. For each value of m, 20 trials were conducted and the sum of the error proxy in time was computed and plotted in the figure below.

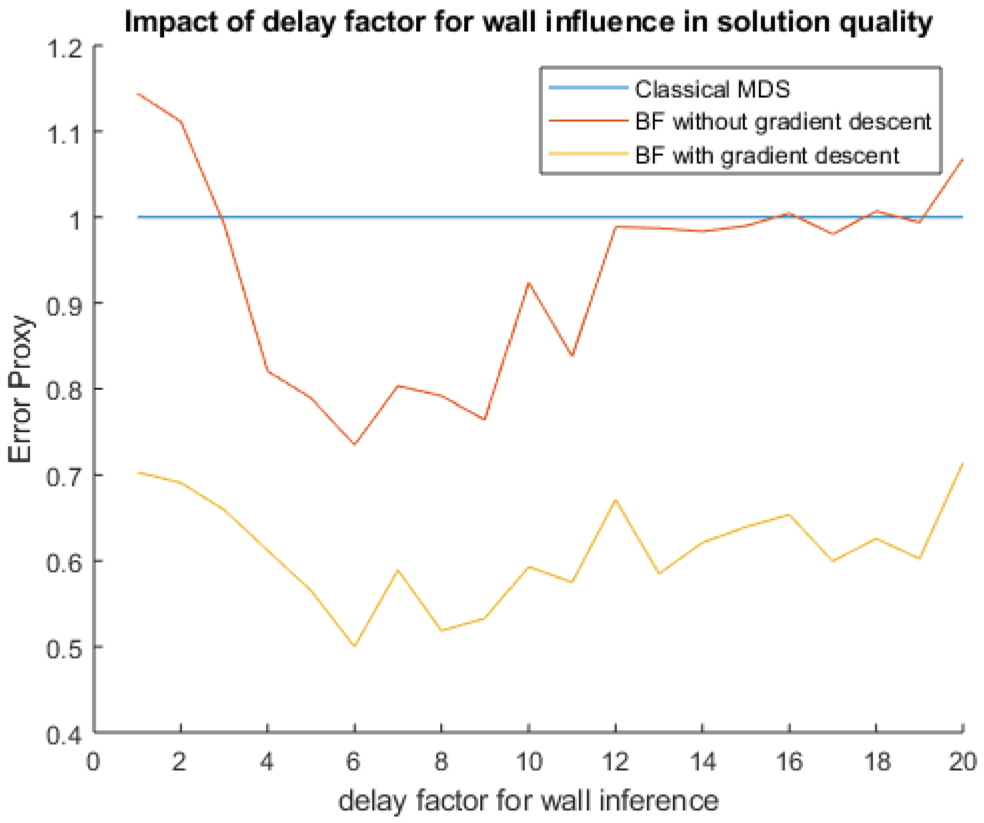
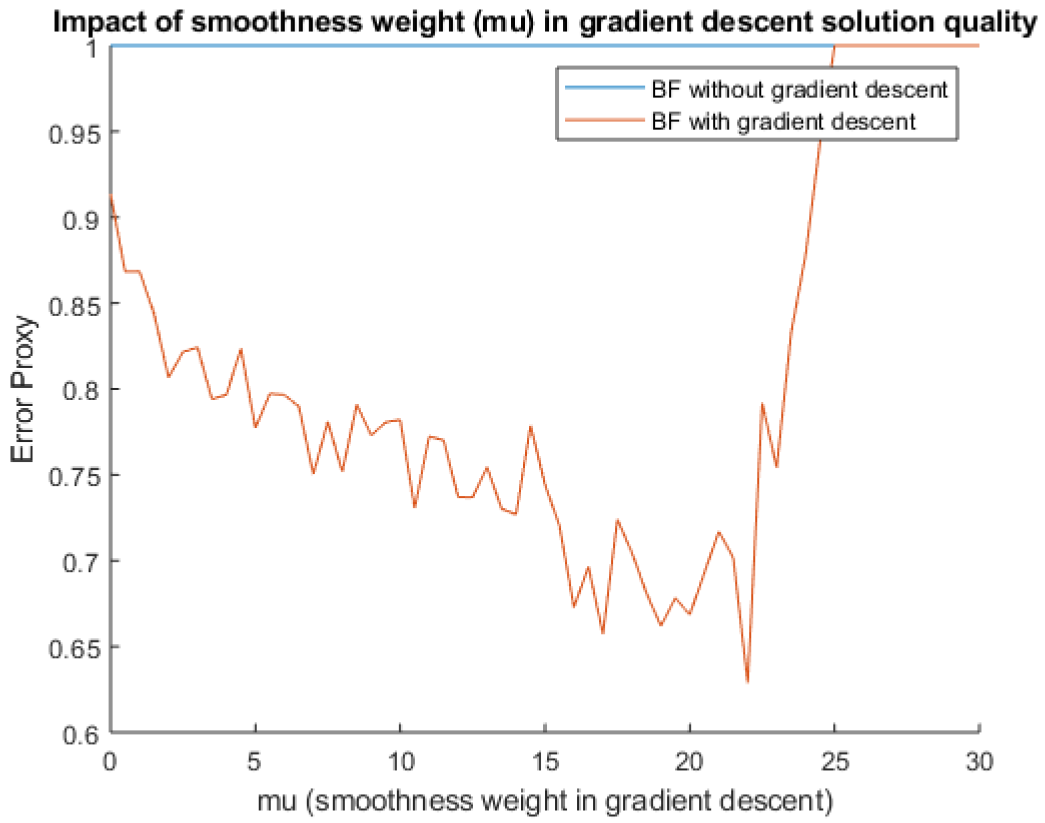


Figure X.

Each point represents the average of 20 trials run for that particular value of m. The Error proxy values were normalized with respect to that of Classical MDS.

The optimal m was empirically chosen to be $m = 6$. For small m, BF performs more poorly than Classical MDS due to somewhat random wall inferences. for $m > 12$, walls are seldom inferred, and as a result BF tends to be approximately the same methodology as Classical MDS.

The gradient used in the gradient descent portion of the algorithm contained a smoothness component with a coefficient mu. BF's performance for various values of mu is shown in the following figure.



In this particular application of BF, a coefficient of 0.01 was multiplied with the gradient to determine the size of the step taken for each coordinate direction during the gradient descent optimization. $\mu = 20$ was chosen as optimal for this purposes of this study.

Multiwall case

BF is useful in environments with multiple walls. Consider the following performance comparison.

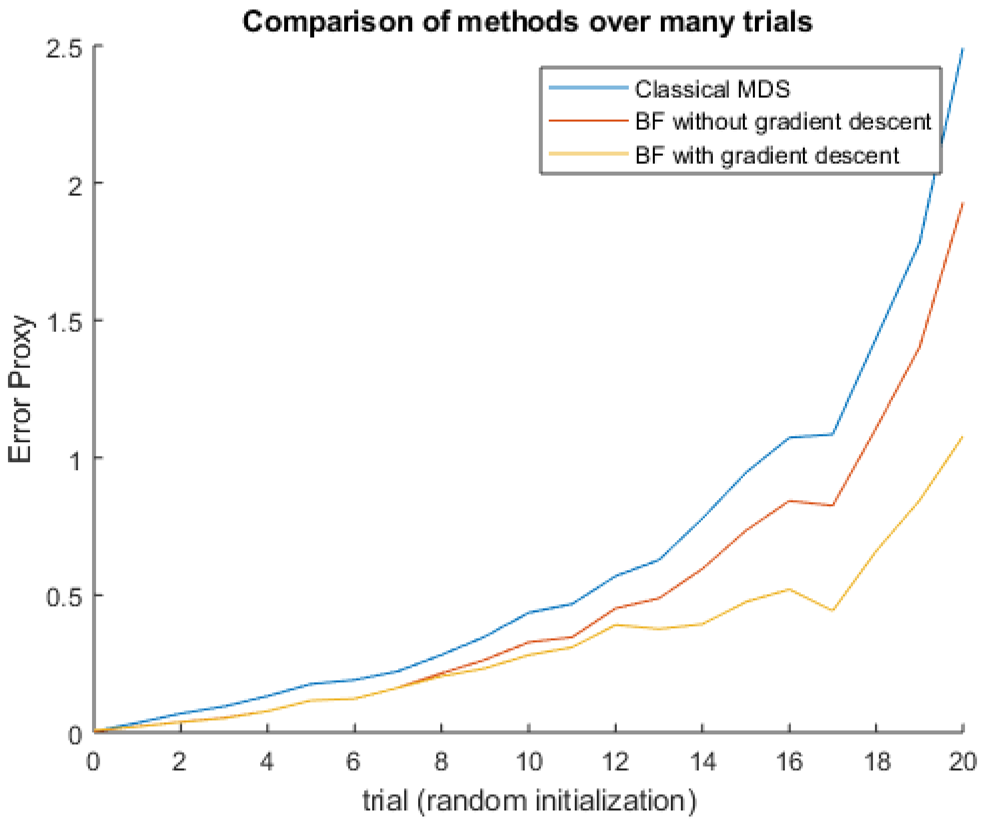


Figure X.

Each point represents the average of 20 trials run for that particular number of walls in the environment.

When the number of walls exceeds 4, the positioning error for BF becomes roughly as poor as the proxy error of Classical MDS in the single wall case. However, it is worth noting that receiving a wireless signal through 4 or more walls is quite rare for most RSSI systems. Instead, nodes separated by such a large number of walls are typically completely out of range and do not even have a contaminated distance measurement. Therefore, in a wall-rich environment, a system would utilize BF to solve for the positions of subsets of the total node network that are influenced by a small number of walls. With sufficient network density, these sub-solutions could combine to form a cohesive network map.

For an environment of 3 walls (and random measurement noise of 15%) the following performance was observed.

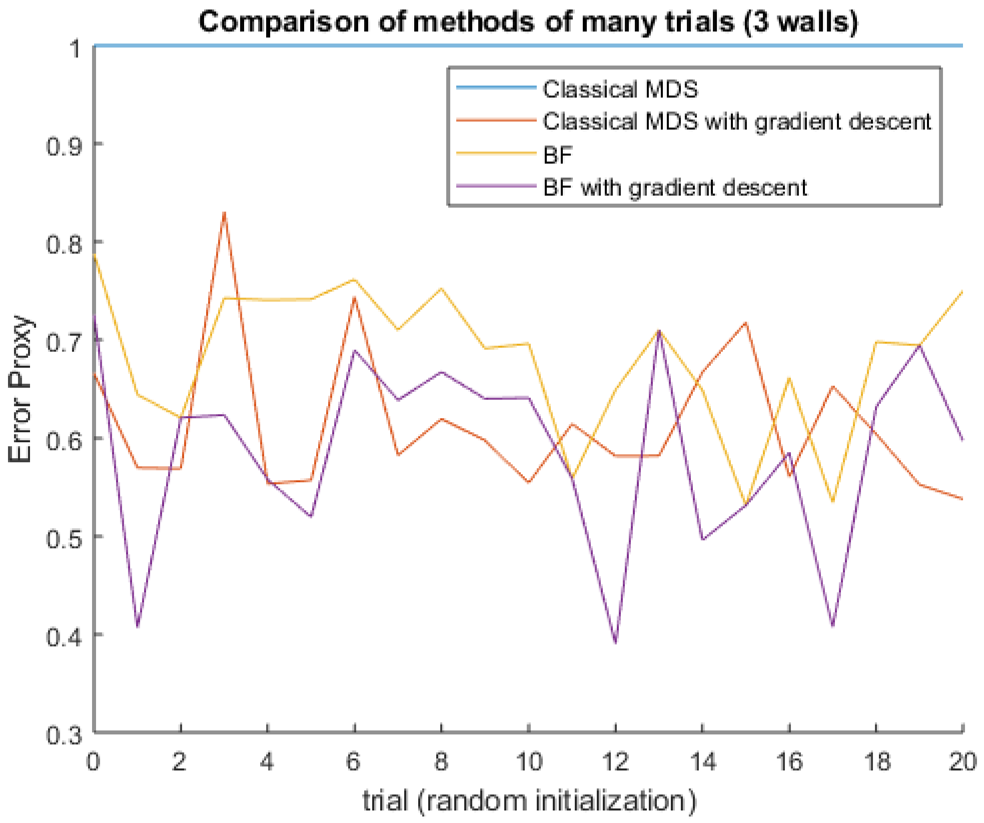


Figure X.
A comparison of the error for each algorithm normalized to the error for Classical MDS over 20 trials.

Extracting a Map of the Environment

The executing of BF generates both a wall-corrected localization map $\mathcal{X}$ and a wall inference tensor $\mathcal{Q}$. The combination of $\mathcal{X}$ and $\mathcal{Q}$ contain some information about the location of the obstacles in the environment. Specifically, if $\mathcal{Q}(i,j,t) > 0$, it is practical to assign a uniform probability density of a portion of an obstacle lying along the line segment from $\mathcal{X}(i,:,t)$ to $\mathcal{X}(j,:,t)$.

Environment Mapping Algorithm

```
for $t = 1 : T$
    for $i = 1 : N$
        for $j = i + 1 : N$
            if $(\mathcal{Q}(i,j,t) > 0)$
                Increment field strength between $\mathcal{X}(i,:,t)$ and $\mathcal{X}(j,:,t)$ by $\mathcal{S}_{\mathcal{X}}(i,j,t)^{-1/2}$
            end
        end
    end
end
```

The resulting field *resembles* a probability density function for walls within the environment, although it itself does not constitute a true probability density function. Still, the field facilitates relatively accurate estimations of the locations and sizes of obstacles and enables rudimentary environmental mapping. Known details about the environment (e.g. othogonal walls) could couple with this pseudo-probability density function to produce better estimations of the environmental map.

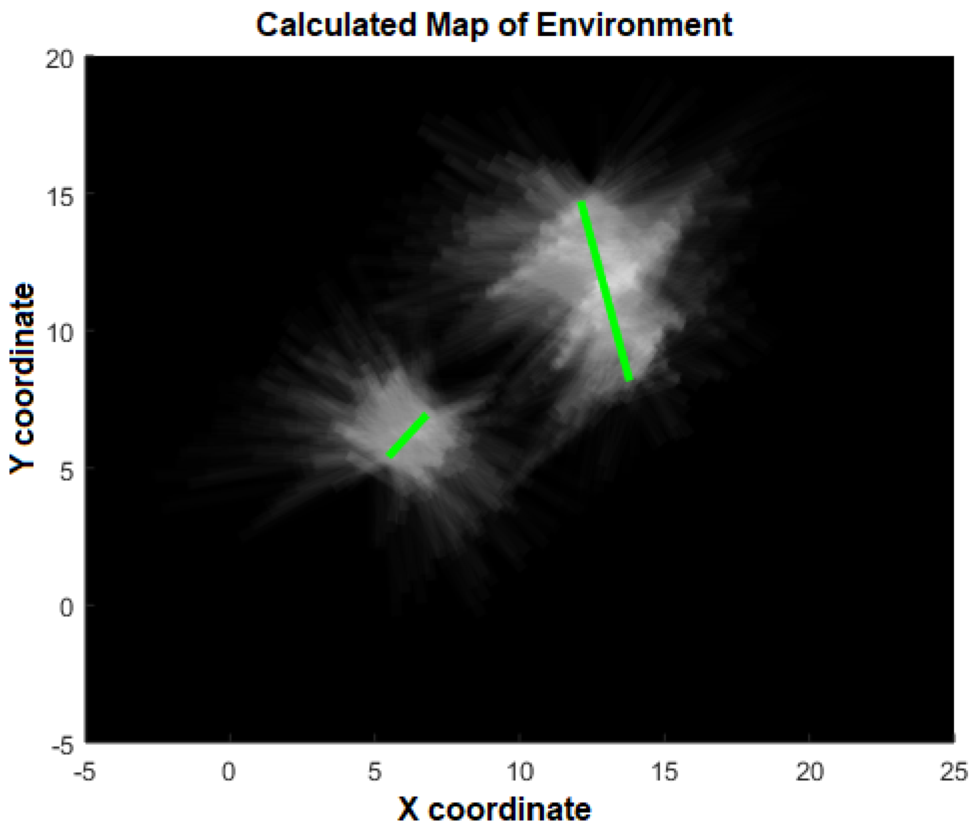


Figure X.
An example of the environment map generated from a simulation run. The real locations of the walls (green) are overlayed on the pseudo-probability density map of obstacle locations (white).

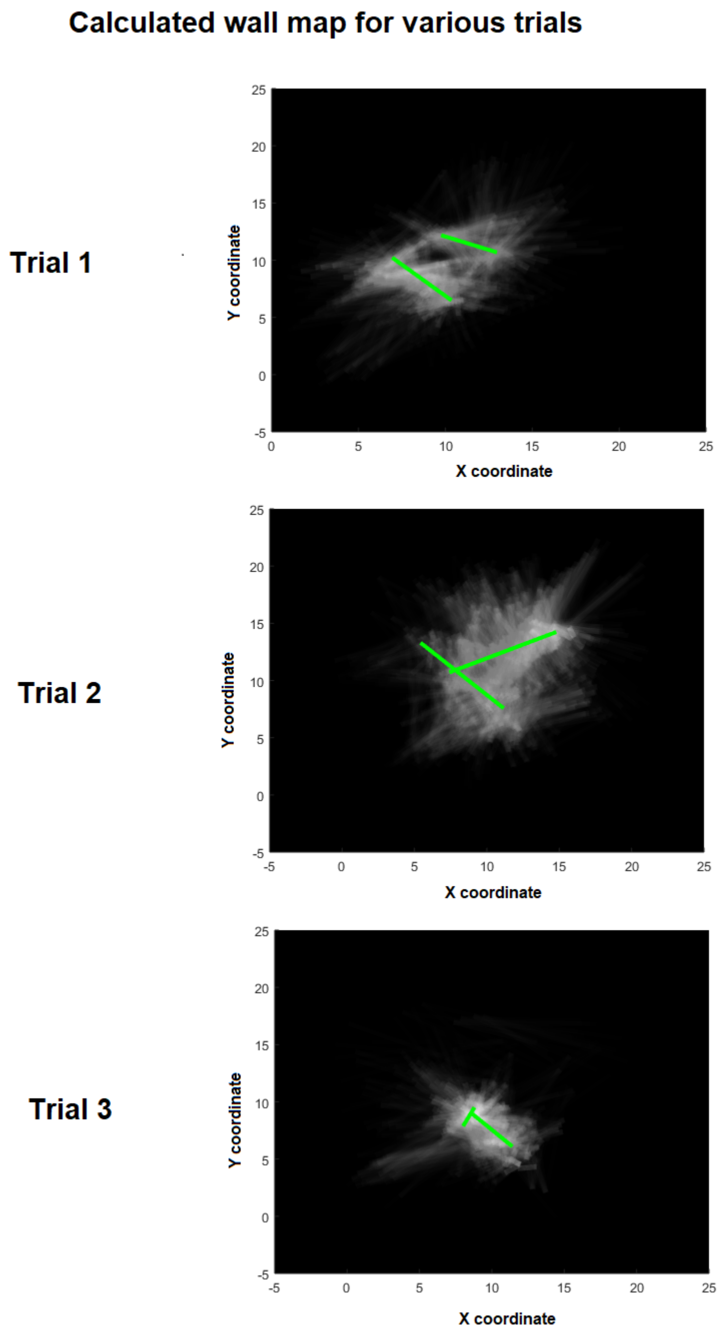


Figure X.
Results from environmental mapping after three different trial runs of the simulation.

Conclusion

I've introduced a "Break Formation" algorithm based on Classical MDS and capable of extending the effectiveness of Classical MDS to environments with significant, systematic errors in node-to-node distance measurements on account of obstacles and walls. Simulations revealed BF yielded node maps with an average error on each node coordinate of less than 5% of the RMS of the node-to-node distances. Thus, BF yielded a percent error in calculated node locations that was 30% lower than the percent error obtained from Classical MDS. Furthermore, I've demonstrated how BF can naturally produce an approximate probability density function of obstacles over the environment space. These results support the use of BF as a localization algorithm to support mapping the locations of personnel teams in an indoor, obstructive environment.

References

[1] B. Li “A Robust Wireless Sensor Network Localization Algorithm in Mixed LOS/NLOS Scenario” Sensors 15 (2015)

[2] C. Miao, et. al. “RI-MDS: Multidimensional Scaling Iterative Localization Algorithm Using RSSI in Wireless Sensor Networks” International Journal of Distributed Sensor Networks Volume 2015, Article ID 6872558 (2015)

[3] B. R. Stojkoska “Node Localization in 3D Wireless Sensors Networks Based on Multidimensional Scaling Algorithm” International Scholarly Research Notices Volume 2014 (2014)

[4] C. Di Franco “Dynamic Multidimensional Scaling with anchors and height constraints for indoor localization of mobile nodes” Robotics and Autonomous Systems 108 28-37 (2018)

[5] Q. Li, et. al. “RMDS: Ranging and multidimensional scaling-based anchor-free localization in large-scale wireless ensor networks with coverage holes” International Journal of Distributed Sensor Networks Vol. 13 (2017)

[6] G. Latsoudas, N. D. Sidiropoulos “A Fast and Effective Multidimensional Scaling Approach for Node Localization in Wireless Sensor Networks” IEEE Transactions on Signal Processing Vol. 55 No. 10 (2007)