Modeling and Control of Cyberphysical Systems
Project 1

Daniele Bigagli Federico Maresca Salvatore Sipione

July 2021

1 Introduction

In this project, we will simulate indoor localization and tracking algorithms used for Cyberphisical Systems implemented through a Wireless Sensor Network (WSN) where each sensor acquires the Received Signal Strength (RSS) of a signal broadcast by a target that must be located. Given a physical setting, we will analyze the behaviour of two different localization methods: a centralized localization method, using the Iterative Soft Thresholding (IST) algorithm, and a distributed localization method, using a Distributed Iterative Soft Thresholding (DIST) algorithm.

Then, we will perform a tracking simulation where the target is moving between adjacent cells randomly every consecutive time instant. Running the experiments several times, we can analyze the success rate of the localization/-tracking of the target reference point. All code snippets for the main function are given at the end of the paper and referenced as listings.

2 Physical Setting

The environment setting of this project is described by a square room of 100 m^2 composed of 100 square cells of 1 m^2 each; the reference point of each cell is located at the center. The RSS model is the one defined by IEEE 802.15.4 indoor empirical model:

$$RSS(d) = \begin{cases} P_t - 40.2 - 20\log d + \eta & d \leq 8 \text{ m} \\ P_t - 58.5 - 33\log d + \eta & d > 8\text{m} \end{cases} \tag{1}$$

See code in Listing 1

where $P_t = 25$, η is a Gaussian noise $\eta \sim \mathcal{N}(0, \sigma^2)$, $\sigma = 0.5$. The number of sensors used is 25, while the distribution of these sensors in the environment

is performed in two different ways. The first deployment method is uniformly at random positions, where each sensor is connected with sensors at distance $d \leq 4m$; the second one is by grid topology, where the sensors are deployed on a grid 5 x 5 and connected to 4 closest sensors.

3 Algorithms

3.1 IST: Iterative Soft Thresholding

IST uses a component-wise operator that, is applied at every time step to an input vector x in order to converge to a minimum value of X through proximal gradient descent. Here λ is our threshold shrinkage operator and τ is the time-step for the gradient descent. A is our sensing/predictor matrix and y are measurements. The Soft-Thresholding operator is defined as follows:

$$S_\lambda(x) := \begin{cases} 0 & \text{if } x \leq \lambda \\ x - sgn(x)\lambda & \text{if } x \geq \lambda \end{cases}$$

See code in Listing 4

Algorithm 1: Iterative Soft Thresholding (code in Listing 2)

$x_0 = 0$
for *t= 1, ... , T_{max}* **do**
  $x_t = S_\lambda[x_{t-1} + \tau A^t(y - Ax_{t-1})]$
end

3.2 DIST: Distributed IST

DIST uses IST but, instead of using information known only to the current node of the system, it distributes information through the network described by Q consensus matrix. Every node gathers information from its neighbours before appliying IST to gain knowledge on its own next state.

Algorithm 2: Distributed Iterative Soft Thresholding (code in Listing 3)

for *each i* **do**
  $x_i(0) = 0$
end
for *t= 1, ... , T_{max}* **do**
  for *each i* **do**
    $x_i(t) = S_\lambda[\sum_{j=1}^{n} Q_{i,j}x_j(t-1) + \tau A_i^t(y_i - A_i x_i(t-1))]$
  end
end

Given the following inequality for a non stopped distributed gradiend descent:

$$\|x_i(t) - \bar{x}_i\| \leq \frac{\tau}{1 - esr(Q)} D$$

Yuan, Ling,Yin, 2015

where esr(Q) is the second largest eigenvalue for the matrix Q defining the network in question we can state that convergence time is linked to the esr. In fact the lower the esr is the quicker the convergence time will be. However during our experiments the average DIST convergence time varied between models with different esr without showing any major correlation.

4 Training Phase: RSS fingerprinting

During the training phase, each sensor takes $m = 1$ measurement for each reference point, we calculate the squared distance between that point and the sensor position and then, using the RSS (1) function, we store the signal strength into a global dictionary $A \in \mathbb{R}^{mn,p}$. The matrix A is orthogonalized with the Feng Orthogonalization to reduce the coherence of the dictionary, which is usually large and allows us to relax the conditions for:

$$k \leq \frac{1}{2}(1 + \frac{1}{\mu(A)})$$

Fuchs, 2004-05

Where μ is the coherence of matrix A. This is for small noise and small λ.

5 Runtime Phase

In the first part of the simulation, we implement the two sensor deployment method that can be switched using a dedicated variable; in the grid topology, we put the sensors in the fixed position shown in figure. In the other case, we take a random uniformly distributed position for each sensor and we check the connection of the graph, stated that sensors are connected if the distance between them is $\leq 4m$ and if it's not, we repeat the distribution.

We analyze the results of a total of 50 experiments, run for each topology configuration.

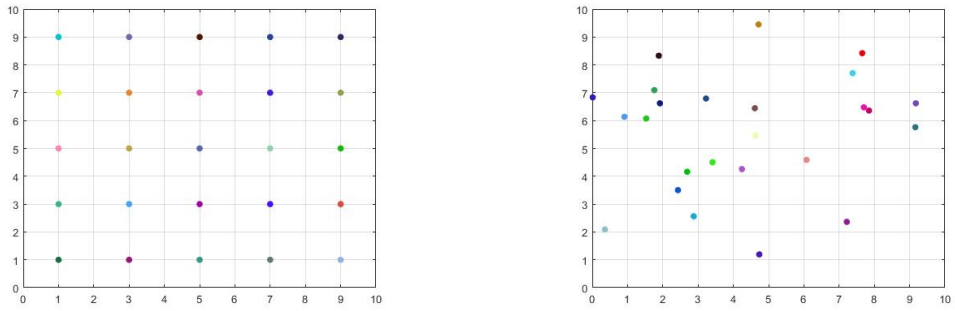


Figure 1: Grid topology distribution (left) and Random topology distribution (right)

5.1 Localization

Each time, the target is positioned in a random point of the room and the two algorithms try to estimate in which cell that point lies. The rate of success tends to be slightly higher with the grid distribution (Figure 2a), scoring a 62% success with the IST algorithm (see Listing 5) and a 58% for the DIST (see Listing 6). With the random deployment, (Figure 3a) the score of both the algorithms is 52%. Due to the random position of the target, that can result near a cell bound, the algorithm often fail just for one adjacent cell, representing a good localization overall. We set a stop criterion for the IST and the DIST defined by:

$$||x(t) - x(t-1)||_2 < 10^{-5} \tag{2}$$

.

5.2 Tracking

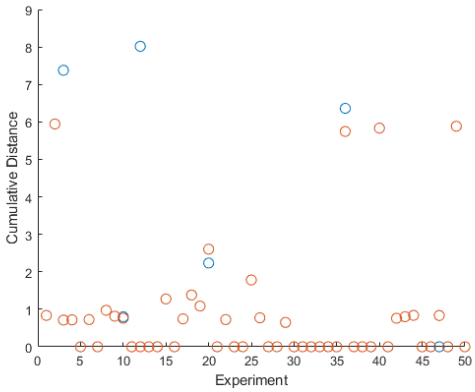
In the tracking simulation, the target is set in a random cell reference point and moved for 10 steps. In each step, the target moves to one of the adjacent cells, taking a randomly chosen vertical and horizontal direction, whilst making sure that the target will stay inside the bounds. We measure the distance between the reference point of the predicted cell found with the O-DIST algorithm and the actual position of the target in each time step, and we calculate the cumulative distance of the whole tracking experiment:

$$\text{dist} = ||x(t) - x_{O-DIST}(t)||_2, \quad t = 1, ..., T \tag{3}$$

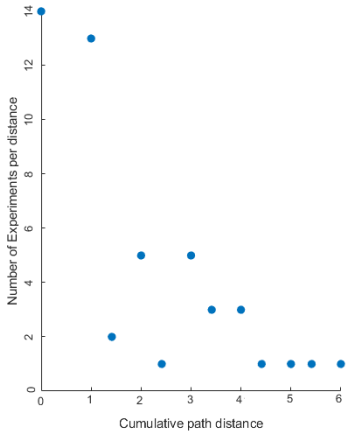
$$\text{exp\_dist} = \sum_{h=1}^{t} ||x(h) - x_{O-DIST}(h)||_2, \quad t = 1, ..., T \tag{4}$$

In the majority of the experiments, shown in Figure 2b the tracking performs well with the grid topology, 14 of the sxperiments make 0 mistakes over the 10

steps and another 13 make just 1, deviating from the true path by 1 meter in total. Only 8 experiments deviate from the path by more than 4 meters. Even when the estimated path deviates a lot in the cumulative distance, at each time step estimated cell is always adjacent to the target cell. The random sensor distribution Figure 3b is less stable in the tracking problem, resulting in a higher cumulative distance through the different experiments. The failed predictions are more frequent over the 10 steps, even though, again, the estimated position is only ever off by one cell with respect to the actual target position.

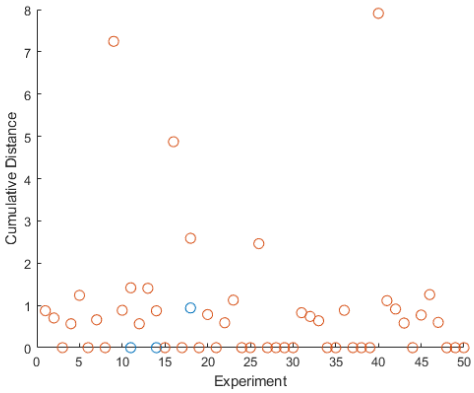


(a) IST (red) and DIST (blue) cumulative distance for each experiment

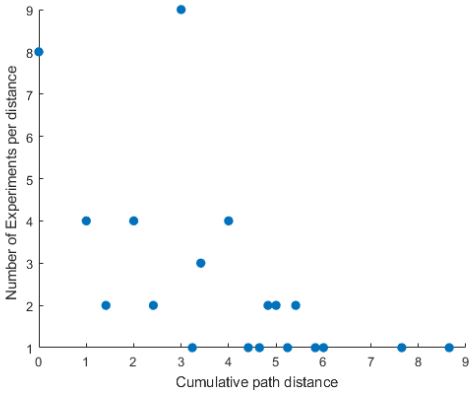


(b) O-DIST: number of experiments with respective cumulative path error

Figure 2: Localization and Tracking error results for the GRID topology



(a) IST (red) and DIST (blue) cumulative distance for each experiment



(b) O-DIST: number of experiments with respective cumulative path error

Figure 3: Localization and Tracking error results for the RANDOM topology

6 CODE SNIPPETS

```
%% Rss Model

function r = RSS(d)
    P_t = 25;
    sigma = 0.5;
    mu = 0;
    noise = sigma*randn+mu;
    if( d <= 8 )
        r = P_t - 40.2 - 20*log10(d) + noise;
    else
        r = P_t - 58.5 - 33*log10(d) + noise;
    end
end
```

Listing 1: RSS

```
%%IST model

function x = IST(A,y,lambda,tau,x_prec)
        x = x_prec + (tau.*A.')*(y-A*x_prec);
        for i=1:length(x_prec)
            x_c = ST(lambda,x(i));
            x(i) = x_c;
        end
end
```

Listing 2: IST

```
%%DIST model

function x = DIST(A,y,lambda,tau,n,p,Q, x_prec)
x=x_prec;
for i=1:n
    xi_new = (Q(i,:)*x')' +(tau.*A(i,:).')*(y(i)-A(i,:)*x(:,i));
                %Qij*xj     %tau*Ai          %yi    Ai*xi
    for j=1:p
      xi_new(j) = ST(lambda,xi_new(j));
    end
    x(:,i) = xi_new;
end
```

Listing 3: DIST

```
%% SOFT THRESHOLDING

function a = ST(lambda,x)
    if( abs(x) <= lambda )
        a = 0;
    else
        a = x-sign(x)*lambda;
    end
end
```

Listing 4: Soft Thresholding

```
%%Outer IST

function [x,t] = outerIST(A,y,lambda,tau,p)
x_prec = zeros(p,1);

    for t =1:1000000
        x = IST(A,y,lambda,tau,x_prec);
        %pruning for loop when converging to lower values
        if (norm(x-x_prec) < 1e-5)
            x_prec = x;
            break;
        end
        x_prec = x;
    end
end
```

Listing 5: Outer loop for IST

```
%%Outer DIST

function [x,t] = outerDIST(A,y,lambda,tau,Q,n,p,x_prec,Tmax)
isOnline = (Tmax > 0 );
for t =1:1000000
        %run DIST at each time step for every sensor
        x = DIST(A,y,lambda,tau,n,p,Q,x_prec);

        %pruning for loop
        if (norm(x-x_prec) < 1e-5 || (isOnline && t > Tmax) )
            x_prec = x;
            break;
        end
        x_prec = x;
end

end
```

Listing 6: Outer loop for DIST

References