APPENDIX A:

Example 1—CDB Filtering

```
clear; clc;

% Structure commom to both reference and target RF fingerprints
% 1st column - Cell ID | 2nd column - RSS | 3rd column - RTT

% Target RF fingerprint
F = [100 62 0; 110 60 -1; 5 54 -1; 2 43 -1; 99 40 -1];
% Reference RF fingerprints
A{1,1} = [100 55 1; 5 50 -1; 110 49 -1; 111 45 -1; 10 34
-1; 200 30 -1; 201 29 -1];
A{1,2} = [100 60 0; 110 50 -1; 2 45 -1; 5 40 -1; 10 35 -1];
A{1,3} = [100 59 1; 110 49 -1; 2 50 -1; 5 39 -1; 10 36 -1];
A{2,1} = [100 54 0; 5 50 -1; 110 49 -1; 111 45 -1; 10 34
-1; 200 30 -1; 201 29 -1];
A{2,2} = [100 61 0; 110 50 -1; 2 45 -1; 5 40 -1; 10 35 -1];
A{2,3} = [110 60 0; 2 52 -1; 100 50 -1; 5 39 -1];
A{3,1} = [110 63 0; 2 52 -1; 100 50 -1; 5 38 -1];
A{3,2} = [110 60 0; 100 52 -1; 2 50 -1];
A{3,3} = [110 59 1; 100 52 -1; 2 50 -1];

N = 4; % Parameter N - Particular Case

k = 0; % Initialize B set index
for i=1:size(A,1)
     for j=1:size(A,2)
          % First Filtering Step
          if A{i,j}(1,1)==F(1,1)
            k=k+1;
            % B set stores the fingerprint position
            (line,column) within A
            B{k}=[i j]; %#ok<AGROW>
      end;
    end;
end;

k = 0; % Initialize C set index
for i=1:size(B,2)
     % Second Filtering Step
     if A{B{i}(1),B{i}(2)}(1,3)==F(1,3)
        k=k+1;
        % C set stores the fingerprint position (line,column)
        within A
        C{k}=[B{i}(1),B{i}(2)]; %#ok<AGROW>
     end;
end;
```

```
% Set of the N cell IDs with the highest RSS values in
target fingerprint
ITn = F(1:N,1)';

k = 0; % Initialize D set index
for i=1:size(C,2)
     % Set of cell IDs in the reference RF fingerprint at
     current pixel
     IRij = A{C{i}(1),C{i}(2)}(:,1)';
     % Third Filtering Step
     if size(intersect(ITn,IRij),2)==N
       k=k+1;
       D{k}=[C{i}(1),C{i}(2)]; %#ok<AGROW>
     end;
end;

display('Search Space Reduction Factor');
disp(1-size(D,2)/(size(A,1)*size(A,2)));
```

APPENDIX B:

Example 2—RSS Correlation

```
clear; clc;

% Structure commom to both reference and target RF fingerprints
% 1st column - Cell ID | 2nd column - RSS | 3rd column - RTT

% Matrix F - target RF fingerprint
F = [100 62 0; 110 60 -1; 5 54 -1; 2 43 -1];

% Matrix Sa - reference RF fingerprint at pixel (10,20)
Sa = [100 55 0; 5 50 -1; 110 49 -1; 111 45 -1; 10 34 -1;
200 30 -1];

% Matrix Sb - reference RF fingerprint at pixel (10,25)
Sb = [100 60 0; 110 50 -1; 2 45 -1; 5 40 -1; 10 35 -1];

N = 3; % Parameter N - Particular Case
Delta = 6; % Parameter Delta (dB)
Beta = 63; % RSS dynamic range (GSM test)
Na = size(F,1); % Number of Anchor Cells = number of lines
if F

% Initialize distance vector
D(1) = 0; % Particular Case, using Euclidean Distance
D(2) = 0; % Generic Case with Penalty Term, using SAD

% Initialize number of equivalent sectors (used for the
penalty term
```

```
% calculation in the Generic Case)
NumEq = 0;
S = Sa; % Selects which reference fingerprint is being used
(Sa or Sb)

for i=1:Na
     % Returns the line in S where the F i-th cell ID appears
     Line = find(S(:,1)==F(i,1));
     if i<=N % Particular case - only the N strongest
        % Particular Case, using Euclidean Distance
        D(1) = D(1) + floor(abs(S(Line,2)-F(i,2))/Delta)^2;
     end;
     % Generic Case with Penalty Term, using SAD
     if ~isempty(Line) % if F(i,1) is listed in the
     reference fingerprint S
       D(2) = D(2) + floor(abs(S(Line,2)-F(i,2))/Delta);
       NumEq = NumEq+1;
     end;
end;

D(1)=sqrt(D(1)); % Euclidean Distance
D(2)=D(2)+2*Beta*(Na-NumEq); % Adds Penalty Term

% Displays the current reference fingerprint and the results
display('Reference RF fingerprint');disp(S);
display('Particular Case, using Euclidean Distance');
disp(D(1));
display('Generic Case with Penalty Term, using
SAD');disp(D(2));
```

APPENDIX C:

Example 3—Part 1: ANN Training

```
% this example requires:
% - MATLAB version 2008a or higher
% - Neural Network Toolbox
clear; clc; close all;

% trainset is a MxN matrix, where N is the number of input
patterns and M
% is equal to (2*S+1); S is the number of cells in the test area
load TrainSet;
N = size(TrainSet,2);

% targets is a 2xN matrix, where N is the number of input
patterns in the
% training set; each target is the "true" MS location (X,Y)
load TargetSet;
```

```
% normalized targets for output range compatible with output
layer neurons
% transfer functions
TargetSetNorm = mapminmax(TargetSet);

% replicates input patterns to reinforce ANN learning
% 1st set - direct sequence of input patterns
% 2nd set - random sequence of input patterns
% 3rd set - direct sequence of input patterns
randVec = randperm(N);
TrainSet = [TrainSet TrainSet(:,randVec) TrainSet];
TargetSetNorm = [TargetSetNorm TargetSetNorm(:,randVec)
TargetSetNorm];
% creates backpropagation network
% 15 neurons in the hidden layer
% 'tansig' is the transfer function of the hidden and output
layers neurons
% 'trainlm' is the network training function
net=newff(TrainSet,TargetSetNorm,15,{'tansig','tansig'},'
  trainlm');

% ANN training session parameters
% training performance function
net.performFcn = 'mse';
% performance function goal
net.trainParam.goal= 1e-6;
% maximum number of epochs
net.trainParam.epochs= 100;
% maximum number of validation failures
net.trainParam.max_fail=6;
% patterns selected for training and validation sets are
randomly selected
net.divideFcn='dividerand';
% percentage of patterns used for training (prevents
overtraining)
net.divideParam.trainRatio=0.95;
% percentage of patterns used for training validation
(prevents overtraining)
net.divideParam.valRatio=0.05;
% percentage of patterns used for testing
net.divideParam.testRatio=0;

% run ANN training session
% net - ANN
% tr - training record
[net,tr]=train(net,TrainSet,TargetSetNorm);
```

APPENDIX D:

Example 3—Part 2: ANN Testing

```
% this example requires:
% - MATLAB version 2008a or higher
% - Neural Network Toolbox
clear all; close all; clc;

% testset is a MxN matrix, where N is the number of input
patterns and M
% is equal to (2*S+1); S is the number of cells in the test area
load TestSet;

% targets is a 2xN matrix, where N is the number of input
patterns in the
% test set; each target is the "true" MS location (X,Y)
load TestSetTargets;

% loads ANN (previously trained)
load net;

% feeds the testset patterns to the trained ANN
% the outputs are normalized
XYnorm = sim(net,TestSet);

% de-normalize ANN outputs
for i=1:2
     Max(i)=max(Targets(i,:)); %#ok<AGROW>
     Min(i)=min(Targets(i,:)); %#ok<AGROW>
     XY(i,:)=Min(i)+(XYnorm(i,:)+1)*(Max(i)-Min(i))/2;
     %#ok<AGROW>
end;

% estimates MS positioning error, given by the 2D Euclidean
distance
% between the ANN output and the target
for j=1:size(XY,2)
     LocationError(j)=sqrt((XY(1,j)-Targets(1,j))^2+(XY(2,j)-
     Targets(2,j))^2); %#ok<AGROW>
end;
```

APPENDIX E:

Example 4—Ranking Correlation

```
% This example requires MATLAB Statistics Toolbox
clear; clc;

% Structure commom to both reference and target RF fingerprints
% 1st column - Cell ID | 2nd column - RSS | 3rd column - RTT
```

```
% Matrix F - target RF fingerprint
% NOTE: the lines are classified in descending order of RSS
F = [100 62 0; 110 60 -1; 5 54 -1; 2 43 -1; 99 40 -1];

% Number of anchor cells
Na = size(F,1);

% Matrix S - reference RF fingerprint
% NOTE: the lines are classified in descending order of RSS
S = [100 61 0; 110 50 -1; 2 45 -1; 5 40 -1; 10 35 -1];

% The full list of cell IDs in the test area
V = [1 2 5 10 99 100 110 120 130 200]';
% Number of cells in the test area
Nc = length(V);

% Initialize VF and VS
for i=1:Nc
    VF(i,:) = [V(i) Nc]; %#ok<AGROW>
    VS(i,:) = [V(i) Nc]; %#ok<AGROW>
end;

% The position of each cell in the RSS ranking in F must be
inserted in the
% second column of the correspondent row in VF.
for i=1:Na
    VF(logical(VF(:,1)==F(i,1)),2)=i; %#ok<AGROW>
end;

% The position of each cell in the RSS ranking in S must be
inserted in the
% second column of the correspondent row in VS.
for i=1:size(S,1)
    VS(logical(VS(:,1)==S(i,1)),2)=i; %#ok<AGROW>
end;

% calculates the ranking correlation between the ordered
sequences VF(:,2) and
% VS(:,2) using the Statistics Toolbox function CORR
D = 1-corr(VF(:,2),VS(:,2),'type','Spearman');
```