# Appendix D

## Multipath Searcher by Using Matlab v6.5

### Program Description

Input:

- The total received signals after passing through the multipath channel model.

Output:

- The power delay profile.
- Time delays.

```
%%% Define variable %%%
noise = 9; % noise in dB

%%% Received Total Signals %%%
%%% Read Rx I & Q Multipath Rayleigh signals %%%
%% user #1 %%
I = dlmread('C:\12.2kbps\Case6\Idata.txt'); Q =
dlmread('C:\12.2kbps\Case6\Qdata.txt');
%I = ones(1536000,1);
%Q = ones(1536000,1);
%% user #2 %%
%I_2 = dlmread('C:\12.2kbps\Idata.txt');
%Q_2 = dlmread('C:\12.2kbps\Qdata.txt');

%complex_form = (I-j*Q)'+(I_2-j*Q_2)'; % 2 users
complex_form = (I-j*Q)'; % 1 user

%%% Add noise %%%
complex_form = awgn(complex_form,noise);

tap = 6; % # pilot bits correlation

%%% Total considering frame %%%
frame_1 = []; frame_2 = []; frame_3 = []; frame_4 = []; frame_5 =
[]; frame_6 = [];

%%% Pick up 1 from 4 samples (check delay) %%%
%% change 'complex_form' --> 'complex_noise' when add noise %%

%% frame # 1 %%
for n = 0:499
    frame_1(n+1,:) = complex_form(1,1+n:4:256*tap*4+n);
end

%% frame # n %%%
frame_no = 2; for n = 0:499
    frame_2(n+1,:) = complex_form(1,(frame_no-1)*38400*4+1+n:4:(frame_no-1)*38400*4+256*tap*4+n);
end frame_no = 3; for n = 0:499
    frame_3(n+1,:) = complex_form(1,(frame_no-1)*38400*4+1+n:4:(frame_no-1)*38400*4+256*tap*4+n);
end frame_no = 4; for n = 0:499
    frame_4(n+1,:) = complex_form(1,(frame_no-1)*38400*4+1+n:4:(frame_no-1)*38400*4+256*tap*4+n);
end frame_no = 5; for n = 0:499
    frame_5(n+1,:) = complex_form(1,(frame_no-1)*38400*4+1+n:4:(frame_no-1)*38400*4+256*tap*4+n);
end frame_no = 6; for n = 0:499
    frame_6(n+1,:) = complex_form(1,(frame_no-1)*38400*4+1+n:4:(frame_no-1)*38400*4+256*tap*4+n);
end

%%%%%%%%%%%%%%%%%%%%%%%
%%% Matched filter %%%
%%%%%%%%%%%%%%%%%%%%%%%

%% Scrambling code but use OVSF code for Q only %%
OVSF_Q = dlmread('C:\MATLAB6p5\work\ovsf_256_0.txt');

pilot_slot_0 = [1 1 1 1 1 0]; sfq = 256; scramb = []; N = 10;

for n = 1:N
    scramb = [scramb c_long2];
end

corr_q = -2*[pilot_slot_0]+1;
coef_q = corr_q'*OVSF_Q;
```

```
re_corr_q = reshape(coef_q',1,sfq*tap);
%xxx = (re_corr_q).*imag(scramb(1,1:sfq*tap));
MF = (re_corr_q).*conj(scramb(1,1:sfq*tap)); %((re_corr_q).*
%MF = (re_corr_q);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Start Path Searcher %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%

result_1 = []; result_2 = []; result_3 = []; result_4 = [];
result_5 = []; result_6 = [];

%%% Correlation (have to repeat 'frame_no' times)%%%
%% Frame # n %%
for count = 1:500
    correlate = sum(MF.*frame_1(count,:))/(sfq*tap);
    result_1 = [result_1 correlate];
end for count = 1:500
    correlate = sum(MF.*frame_2(count,:))/(sfq*tap);
    result_2 = [result_2 correlate];
end for count = 1:500
    correlate = sum(MF.*frame_3(count,:))/(sfq*tap);
    result_3 = [result_3 correlate];
end for count = 1:500
    correlate = sum(MF.*frame_4(count,:))/(sfq*tap);
    result_4 = [result_4 correlate];
end for count = 1:500
    correlate = sum(MF.*frame_5(count,:))/(sfq*tap);
    result_5 = [result_5 correlate];
end for count = 1:500
    correlate = sum(MF.*frame_6(count,:))/(sfq*tap);
    result_6 = [result_6 correlate];
end

%%% Average all frames (in term of power)%%%
avg = (abs(result_1).^2+abs(result_2).^2+abs(result_3).^2+
abs(result_4).^2+abs(result_5).^2+abs(result_6).^2)/6;

plot(avg); grid; hold;

thres = 4*mean(avg)*ones(1,500); plot(thres,'-r')

%%% Peak search (probability of peak)%%%
power = avg;

%% This part check whether 5 arrays are peak or not, if yes it will check #sample as well %%
sample_search = []; % not necessary, can be removed
peak = []; peak_search = [];

count = 0; for n = 1:496
    if power(1,n) <= power(1,n+1) & power(1,n+3) >= power(1,n+4)
        sample_search = [sample_search;n]; % count sample that may be a peak
        profile_array = [power(1,n) power(1,n+1) power(1,n+2) power(1,n+3) power(1,n+4)];
        [y,x] = max(profile_array);
        peak = [peak;[y,x]];
        x = x+n-1;
        peak_search = [peak_search;[y,x]]; %% power %% sample %%
        count = count+1;
    end
end

%% Find 1-st peak (max peak)%%
power_delay = peak_search(1,:); for n = 1:count-1
    if power_delay > peak_search(n+1,:)
```

```
        power_delay = power_delay;
    elseif power_delay <= peak_search(n+1,:)
        power_delay = peak_search(n+1,:);
    end
end path_1 = power_delay

%% Removed maximum peak %%
count_2 = 0; remain_peak_1 = []; for m = 1:count
    if peak_search(m,1) < path_1(1,1)
        peak_2 = peak_search(m,:);
        count_2 = count_2+1;
        remain_peak_1 = [remain_peak_1;peak_2];
    end
end

%% Find 2-nd peak %%
power_delay_2 = remain_peak_1(1,:); for n = 1:count_2-1
    if power_delay_2 > remain_peak_1(n+1,:)
        power_delay_2 = power_delay_2;
    elseif power_delay_2 <= remain_peak_1(n+1,:)
        power_delay_2 = remain_peak_1(n+1,:);
    end
end

path_2 = power_delay_2

%% Removed 2-nd peak %%
count_3 = 0; remain_peak_2 = []; for m = 1:count_2
    if remain_peak_1(m,1) < path_2(1,1)
        peak_3 = remain_peak_1(m,:);
        count_3 = count_3+1;
        remain_peak_2 = [remain_peak_2;peak_3];
    end
end

%% Find 3-rd peak %%
power_delay_3 = remain_peak_2(1,:); for n = 1:count_3-1
    if power_delay_3 > remain_peak_2(n+1,:)
        power_delay_3 = power_delay_3;
    elseif power_delay_3 <= remain_peak_2(n+1,:)
        power_delay_3 = remain_peak_2(n+1,:);
    end
end

path_3 = power_delay_3
```