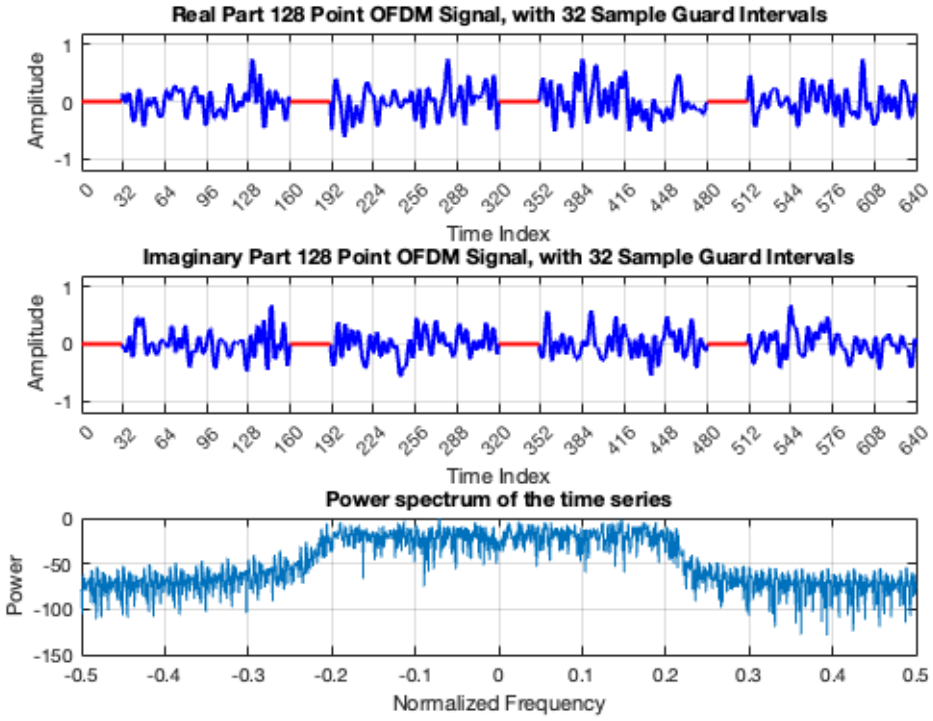
Contents


```
%40 MHz sample rate, 1/40 usec/sample, 160 samples/symbol, 4 usec duration
% 1/40 usec/sample, 128 samples/symbol, 3.2 usec transform segment
% spectral channel spacing 1/3.2 usec = 312.5 kHz

clear all; close all; clc;
```

SECTION A

```
x0=[]; % empty array for OFDM with Guard interval
y0=[]; % empty array for OFDM with cyclic prefices

for k=1:100
    fxx=zeros(1,128);
    dat=(floor(4*rand(1,54))-1.5)/1.5+j*(floor(4*rand(1,54))-1.5)/1.5; %generate random data for 16-QAM
    fxx(65+(-27:27))=[dat(1:27) 0 dat(28:54)];
    xx=ifft(fftshift(fxx));
    xx=0.75*xx/max(xx);
    x0=[x0 zeros(1,32) xx];
    y0=[y0 xx(97:128) xx];
end

figure(1)

    subplot(3,1,1)

        plot(0:640, real(x0(1:641)),'b','linewidth',2)

        hold on
        for k=1:160:640
            plot(k-1:k+30,real(x0(k:k+31)),'r','linewidth',2)
        end
        hold off

        grid on
        set(gca,'XTick', [0:32:640])
        axis([0 640 -1.2 1.2])
        title('Real Part 128 Point OFDM Signal, with 32 Sample Guard Intervals')
        xlabel('Time Index')
        ylabel('Amplitude')

    subplot(3,1,2)

        plot(0:640, imag(x0(1:641)),'b','linewidth',2)

        hold on
```

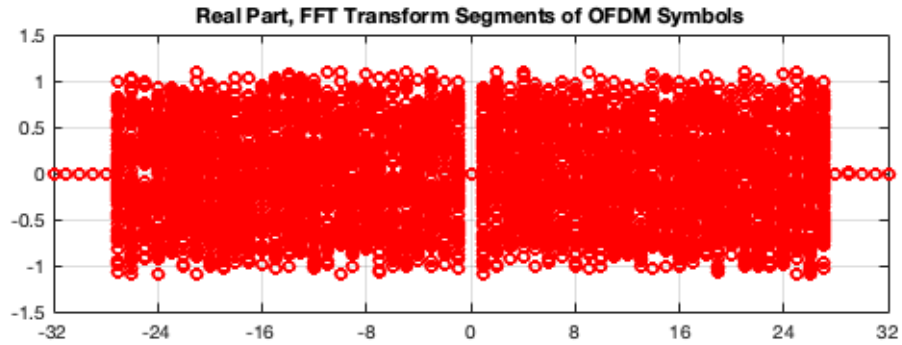
```
    for k=1:160:640
        plot(k-1:k+30,imag(x0(k:k+31)),'r','linewidth',2)
    end
    hold off

    grid on
    set(gca,'XTick', [0:32:640])
    axis([0 640 -1.2 1.2])
    title('Imaginary Part 128 Point OFDM Signal, with 32 Sample Guard Intervals')
    xlabel('Time Index')
    ylabel('Amplitude')


subplot(3,1,3)

    W_k = kaiser(length(x0), 8.3);
    ps = abs(fftshift(fft(x0'.*W_k, 2048))).^2;

    f = linspace(-0.5, 0.5, 2048);
    plot(f, 20*log10(ps/max(ps)));

    grid on
    title('Power spectrum of the time series')
    xlabel('Normalized Frequency')
    ylabel('Power')
```

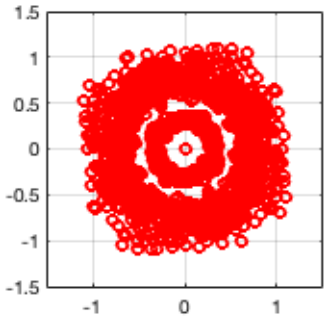


SECTION B

```
figure(2)

    subplot(2,1,1)

        plot(0,0)
```

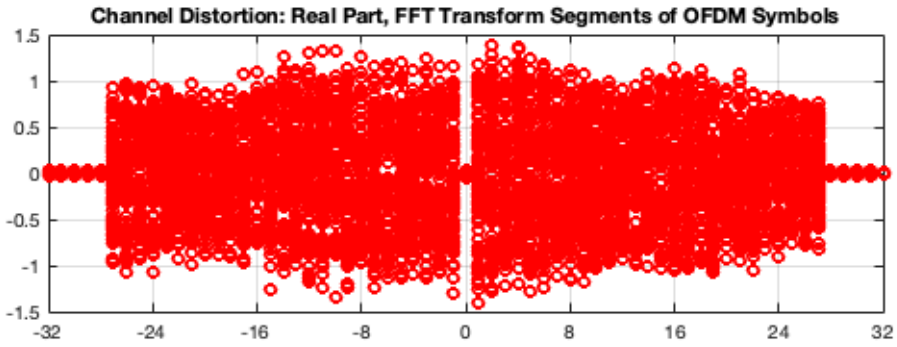
```
    hold on
    for n=1:160:16000-160
        x1=x0(n:n+159);
        fx1=fftshift(fft(x1(33:160)))/8;
        plot(-64:63,real(fx1),'ro','linewidth',2,'markersize',6)
    end
    hold off

    grid on
    axis([-32 32 -1.5 1.5])
    set(gca,'XTick',[-64:8:64])
    title('Real Part, FFT Transform Segments of OFDM Symbols')


subplot(2,1,2)

    plot(0,0)

    hold on
    for n=1:160:16000-160
        x1=x0(n:n+159);
        fx1=fftshift(fft(x1(33:160)))/8;
        plot(fx1,'ro','linewidth',2,'markersize',6)
    end
    hold off

    grid on
    axis('equal')
    axis([-1.5 1.5 -1.5 1.5])
```

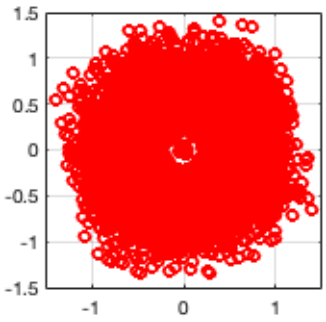




SECTION C

```
x0c = conv(x0, [1   0   0.2   0   0   0   0   0   0   j*0.1]);

figure(3)

    subplot(2,1,1)

        plot(0,0)

        hold on
        for n=1:160:16000-160
            x1=x0c(n:n+159);
            fx1=fftshift(fft(x1(33:160)))/8;
            plot(-64:63,real(fx1),'ro','linewidth',2,'markersize',6)
        end
        hold off

        grid on
        axis([-32 32 -1.5 1.5])
        set(gca,'XTick',[-64:8:64])
        title('Channel Distortion: Real Part, FFT Transform Segments of OFDM Symbols')


    subplot(2,1,2)

        plot(0,0)

        hold on
        for n=1:160:16000-160
            x1=x0c(n:n+159);
            fx1=fftshift(fft(x1(33:160)))/8;
            plot(fx1,'ro','linewidth',2,'markersize',6)
        end
        hold off

        grid on
        axis('equal')
        axis([-1.5 1.5 -1.5 1.5])
```

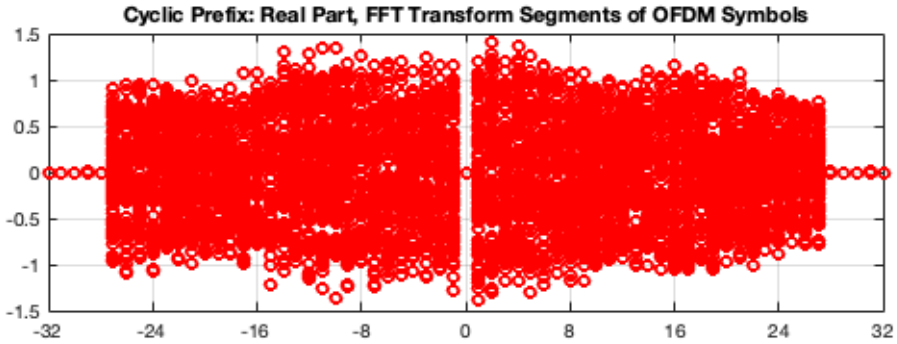


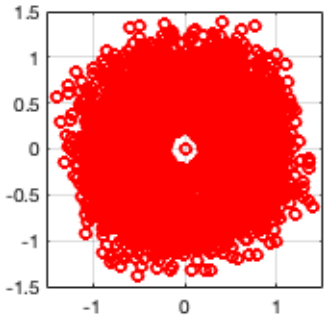


SECTION D

```
y0c = conv(y0, [1    0    0.2    0    0    0    0    0    0    j*0.1]);

figure(4)

subplot(2,1,1)

        plot(0,0)

        hold on
        for n=1:160:16000-160
            y1=y0c(n:n+159);
            fy1=fftshift(fft(y1(33:160)))/8;
            plot(-64:63,real(fy1),'ro','linewidth',2,'markersize',6)
        end
        hold off

        grid on
        axis([-32 32 -1.5 1.5])
        set(gca,'XTick',[-64:8:64])
        title('Cyclic Prefix: Real Part, FFT Transform Segments of OFDM Symbols')


    subplot(2,1,2)

        plot(0,0)

        hold on
        for n=1:160:16000-160
            y1=y0c(n:n+159);
            fy1=fftshift(fft(y1(33:160)))/8;
            plot(fy1,'ro','linewidth',2,'markersize',6)
        end
```

```
        hold off

        grid on
        axis('equal')
        axis([-1.5 1.5 -1.5 1.5])
```

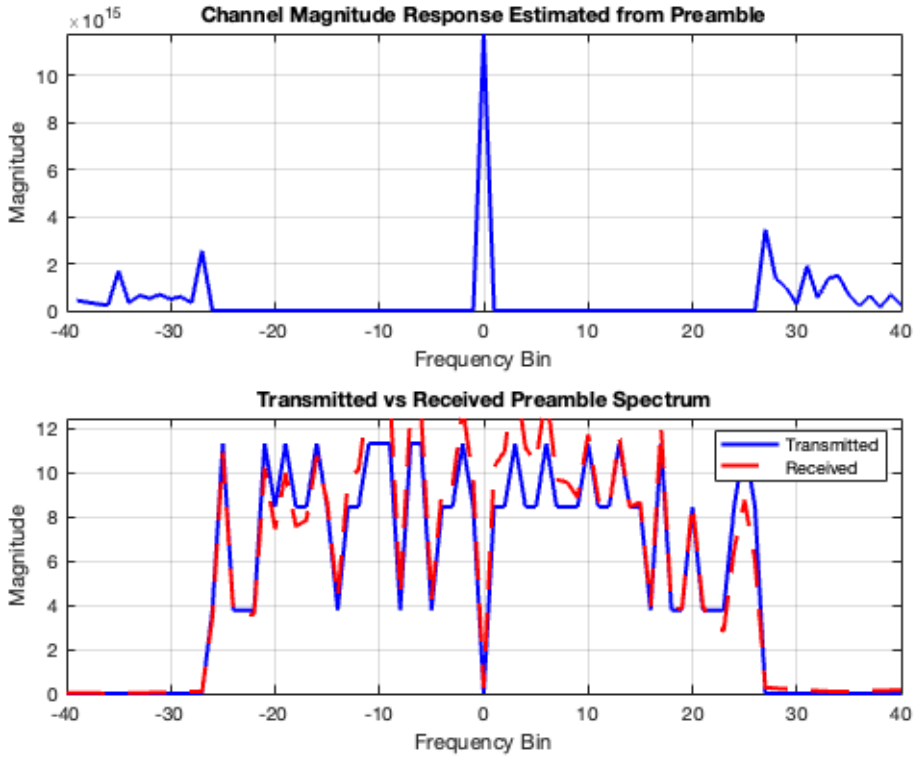




SECTION E

```
P = [];

for k=1:100

    Pf = (floor(4*rand(1,52))-1.5)/1.5+j*(floor(4*rand(1,52))-1.5)/1.5;

    fpp = zeros(1,128);
    fpp(65+[-26:-1 1:26]) = Pf;


    iff = 8*ifft(fftshift(fpp), 128);

    P = [P iff];
end

cp = P(end-63:end);

z0 = [cp P P y0];

z0c = conv(z0, [1   0   0.2   0   0   0   0   0   0   j*0.1]);

prc = fftshift(fft(z0c, 128));
sent = fftshift(fft(P, 128));
```

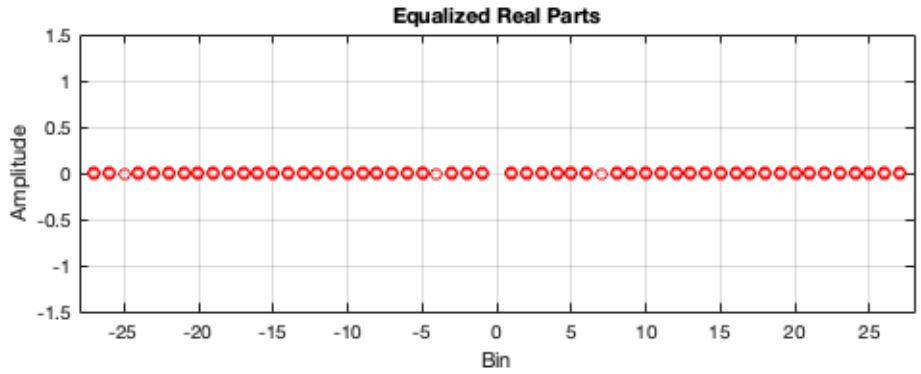
```
figure(5)

    subplot(2, 1, 1)

          % plot channel magnitude response
        channel_response = abs(prc./sent);
        plot(-64:63, channel_response, 'b', 'LineWidth', 2);
        grid on;
        title('Channel Magnitude Response Estimated from Preamble');
        xlabel('Frequency Bin');
        ylabel('Magnitude');
        axis([-40 40 0 max(channel_response)*1.1]);

         subplot(2,1,2)
        % plot received vs transmitted preamble spectrum
        received_preamble = z0c(length(cp)+1:length(cp)+128); % First preamble
        rec_fft = fftshift(fft(received_preamble));
        tx_fft = fftshift(fft(P(1:128)));

        plot(-64:63, abs(tx_fft), 'b', 'LineWidth', 2);
        hold on;
        plot(-64:63, abs(rec_fft), 'r--', 'LineWidth', 2);
        hold off;
        grid on;
        title('Transmitted vs Received Preamble Spectrum');
        xlabel('Frequency Bin');
        ylabel('Magnitude');
        legend('Transmitted', 'Received');
        axis([-40 40 0 max(abs(tx_fft))*1.1]);
```

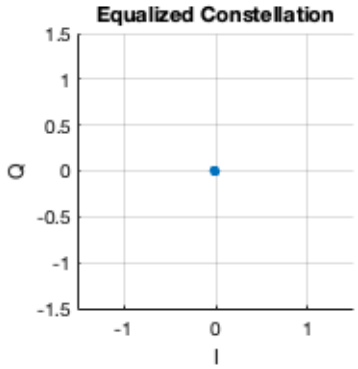


SECTION F: equalizer

extract data portion after preambles

```
data_start = length(cp) + 2*128 + 1;
rx_data = z0c(data_start:end);

% calculate number of complete symbols
n_symbols = floor(length(rx_data)/160);
rx_data = rx_data(1:n_symbols*160); % truncate to integer number of symbols

% reshape into symbols and remove CP
rx_data_reshaped = reshape(rx_data, 160, []);
rx_symbols = rx_data_reshaped(33:end,:); % Remove CP

% FFT and equalization
% assuming prc and sent are vectors
H_est = prc(1:128) ./ sent(1:128);
 % channel estimate from preamble
equalized_symbols = fft(rx_symbols)./H_est.';

% extract data carriers
data_carriers = [equalized_symbols(2:28,:); equalized_symbols(end-26:end,:)];
f_axis = [-27:-1, 1:27];


figure(6)
    subplot(2,1,1)
        plot(f_axis, real(data_carriers(:,1:min(20,end))), 'ro');
        title('Equalized Real Parts');
        xlabel('Bin'); ylabel('Amplitude');
        axis([-28 28 -1.5 1.5]); grid on;

    subplot(2,1,2)
        scatter(real(data_carriers(:)), imag(data_carriers(:)), 'filled');
        title('Equalized Constellation');
        xlabel('I'); ylabel('Q');
        axis([-1.5 1.5 -1.5 1.5]); grid on; axis square;
```





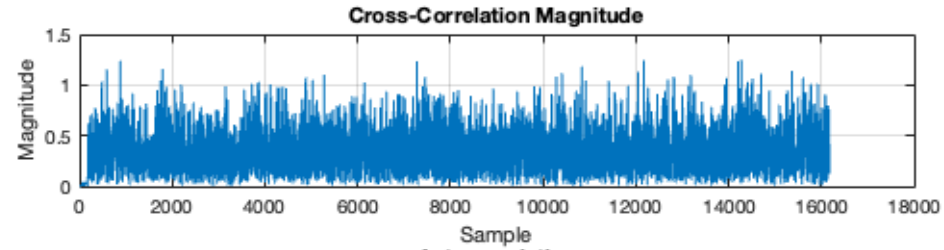
SECTION G

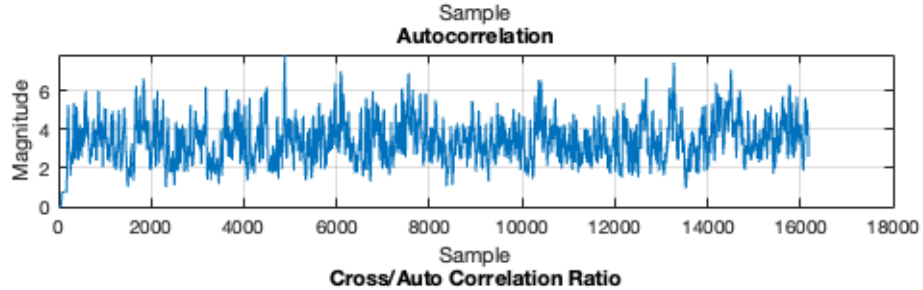
create special preamble with only bins 4,8,...,24

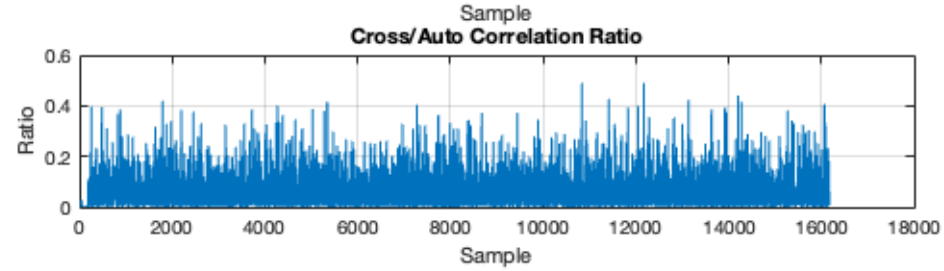
```
special_bins = 4:4:24;
special_preamble_freq = zeros(1,128);
special_preamble_freq(64+1+special_bins) = pskmod(randi([0 3],1,length(special_bins)),4,pi/4,'gray');
special_preamble_time = ifft(ifftshift(special_preamble_freq)) * 8;

% create test signal with delay and rotation
test_signal = [zeros(1,50) special_preamble_time y0];
test_signal_noisy = test_signal + 0.01*(randn(size(test_signal)) + 1j*randn(size(test_signal)));

% add rotation (3 degrees/sample)
rotation_rate = 3; % degrees/sample
rotation = exp(1j*deg2rad(rotation_rate)*(0:length(test_signal_noisy)-1));
test_signal_rotated = test_signal_noisy .* rotation;

% preamble detection using cross-correlation
delay_line = zeros(1,32);
cross_corr = zeros(1,length(test_signal_rotated));
auto_corr = zeros(1,length(test_signal_rotated));

for n = 32:length(test_signal_rotated)
    delay_line = [test_signal_rotated(n) delay_line(1:end-1)];
    cross_corr(n) = sum(conj(delay_line) .* special_preamble_time(1:32));
    auto_corr(n) = sum(abs(delay_line).^2);
end

ratio = abs(cross_corr).^2 ./ (auto_corr * sum(abs(special_preamble_time(1:32)).^2));

% find preamble start
[~, preamble_start] = max(ratio);
```

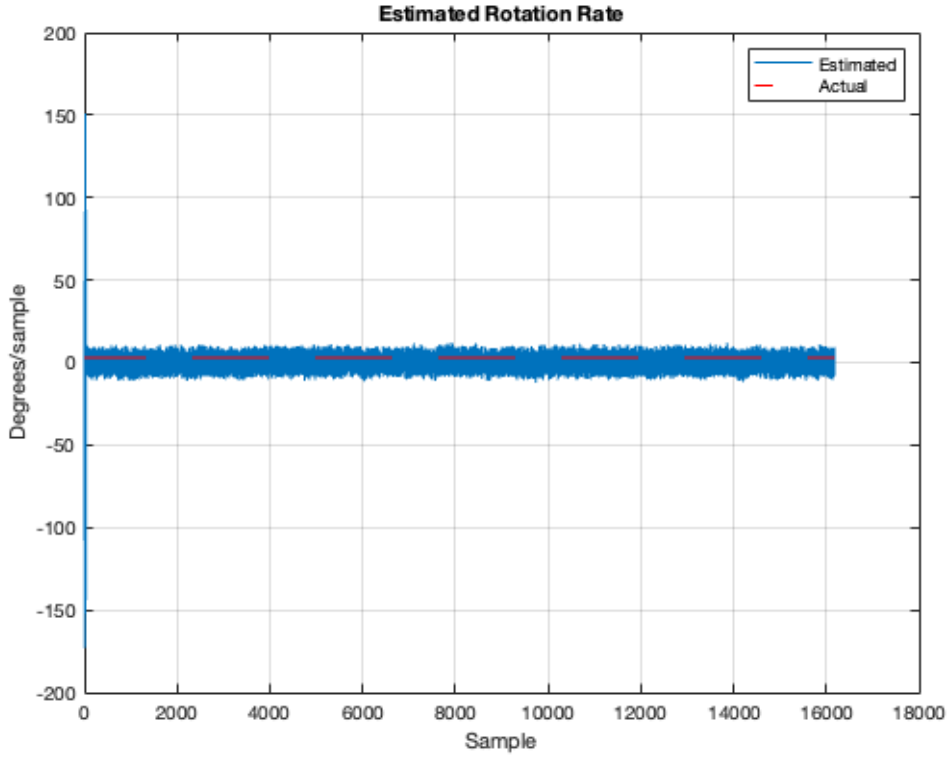
```
% rotation estimation
rotation_est = zeros(1,length(test_signal_rotated));
rotation_est(1:32) = angle(test_signal_rotated(1:32));

for n = 33:length(test_signal_rotated)
    rotation_est(n) = 0.95*rotation_est(n-1) + 0.05*angle(test_signal_rotated(n));
end

% estimate rotation rate
rotation_diff = diff(unwrap(rotation_est));
rotation_rate_est = mean(rotation_diff(preamble_start:preamble_start+32));

figure(7)
    subplot(3,1,1)
        plot(abs(cross_corr));
        title('Cross-Correlation Magnitude');
        xlabel('Sample');
        ylabel('Magnitude');
        grid on;

    subplot(3,1,2)
        plot(auto_corr);
        title('Autocorrelation');
        xlabel('Sample');
        ylabel('Magnitude');
        grid on;

    subplot(3,1,3)
        plot(ratio);
        title('Cross/Auto Correlation Ratio');
        xlabel('Sample');
        ylabel('Ratio');
        grid on;

figure(8)
    plot(rad2deg(rotation_diff));
    hold on;
    plot([1 length(rotation_diff)], [rotation_rate rotation_rate], 'r--');
    title('Estimated Rotation Rate');
    xlabel('Sample');
    ylabel('Degrees/sample');
    legend('Estimated', 'Actual');
    grid on;

disp(['Actual rotation rate: ' num2str(rotation_rate) ' degrees/sample']);
disp(['Estimated rotation rate: ' num2str(rad2deg(rotation_rate_est)) ' degrees/sample']);
disp(['Preamble detected at sample: ' num2str(preamble_start)]);
```

```
Actual rotation rate: 3 degrees/sample
Estimated rotation rate: -0.28042 degrees/sample
Preamble detected at sample: 10833
```









Published with MATLAB® R2023b