

```

function [actual_user_SNR, actual_user_rate, actual_user_SNR_primary,
actual_user_SNR_dual, actual_user_rate_primary, actual_user_rate_dual] =
...
    simulate_ideal_mode_algo_2(W_SNR_ideal, W_Rate_ideal,
elevation_quantized_all, distance_ts_all, channel_trace_map,
allowed_elevs, f_c, Ps)

U = size(elevation_quantized_all, 3);
T = size(elevation_quantized_all, 2);

G = 4; alpha = 0.2; beta = 0.1;
snr_threshold = -5; rate_threshold = 0.05;
low_quality_threshold_snr = 2;
low_quality_threshold_rate = 0.5;

actual_user_SNR          = zeros(U, T);
actual_user_rate         = zeros(U, T);
actual_user_SNR_primary  = zeros(U, T);
actual_user_SNR_dual     = zeros(U, T);
actual_user_rate_primary = zeros(U, T);
actual_user_rate_dual    = zeros(U, T);

prev_assign_snr          = zeros(U, 1);
prev_assign_snr_dual     = zeros(U, 1);
prev_assign_rate         = zeros(U, 1);
prev_assign_rate_dual    = zeros(U, 1);

primary_assign_times = 1:2:T;    % e.g. t = 1, 11, 21, ...
dual_assign_times    = 2:2:T;    % e.g. t = 4, 14, 24, ...

for t = 1:T
    W_snr_ideal = squeeze(W_SNR_ideal(:, :, t));
    W_rate_ideal = squeeze(W_Rate_ideal(:, :, t));

    % --- SNR PRIMARY assignment/hold ---
    if ismember(t, primary_assign_times)
        [assign_snr_primary, ~, ~, ~] =
assign_users_munkres_algo_2(W_snr_ideal, ...
        prev_assign_snr, prev_assign_snr_dual, G, alpha, beta, ...
        snr_threshold, low_quality_threshold_snr, t == 1, false);
        prev_assign_snr = assign_snr_primary;
    else
        assign_snr_primary = prev_assign_snr;
    end

    % --- SNR DUAL assignment/hold ---
    if ismember(t, dual_assign_times)
        [~, assign_snr_dual, ~, ~] =
assign_users_munkres_algo_2(W_snr_ideal, ...
        prev_assign_snr, prev_assign_snr_dual, G, alpha, beta, ...
        snr_threshold, low_quality_threshold_snr, t == 1, true);
        prev_assign_snr_dual = assign_snr_dual;
    else
        assign_snr_dual = prev_assign_snr_dual;
    end

    % --- Rate PRIMARY assignment/hold ---
    if ismember(t, primary_assign_times)

```

```

        [assign_rate_primary, ~, ~, ~] =
assign_users_munkres_algo_2(W_rate_ideal, ...
        prev_assign_rate, prev_assign_rate_dual, G, alpha, beta, ...
        rate_threshold, low_quality_threshold_rate, t == 1, false);
        prev_assign_rate = assign_rate_primary;
    else
        assign_rate_primary = prev_assign_rate;
    end

    % --- Rate DUAL assignment/hold ---
    if ismember(t, dual_assign_times)
        [~, assign_rate_dual, ~, ~] =
assign_users_munkres_algo_2(W_rate_ideal, ...
        prev_assign_rate, prev_assign_rate_dual, G, alpha, beta, ...
        rate_threshold, low_quality_threshold_rate, t == 1, true);
        prev_assign_rate_dual = assign_rate_dual;
    else
        assign_rate_dual = prev_assign_rate_dual;
    end

    for u = 1:U
        % SNR: primary and dual assignments for this user at this time
        v1 = 0; v2 = 0;
        if assign_snr_primary(u) > 0
            v1 = W_snr_ideal(u, assign_snr_primary(u));
        end
        if assign_snr_dual(u) > 0
            v2 = W_snr_ideal(u, assign_snr_dual(u));
        end
        actual_user_SNR_primary(u, t) = v1;
        actual_user_SNR_dual(u, t) = v2;

        % Correct SNR Combination: Linear Sum, then dB
        vals = [v1 v2];
        vals_linear = 10.^(vals(~isnan(vals))/10);
        if isempty(vals_linear)
            actual_user_SNR(u, t) = 0; % No assignment at all
        else
            actual_user_SNR(u, t) = 10*log10(sum(vals_linear));
        end

        % Rate: primary and dual assignments
        r1 = 0; r2 = 0;
        if assign_rate_primary(u) > 0
            r1 = W_rate_ideal(u, assign_rate_primary(u));
        end
        if assign_rate_dual(u) > 0
            r2 = W_rate_ideal(u, assign_rate_dual(u));
        end
        actual_user_rate_primary(u, t) = r1;
        actual_user_rate_dual(u, t) = r2;
        actual_user_rate(u, t) = r1 + r2;
    end
end
end

```