```
% Returns a cell array with each cell containing the list of detected
% particle coordinates for each simulated cycle.
%% Function:
function colist_test = coordinatelist(lambda,phi,Tmax,cycles,quantum_efficiency,↙
dark_rate,blur,phys,Y_bias,X_bias)
    % Call the divided probability list from problist.m
    [p,event] = problist(lambda,phi,Tmax,phys,Y_bias,X_bias);
    % Lists of each event for each trial:
    ys = zeros(cycles,1); xs = ys; ws = ys; zs = ys;
    % Find what event each trial corresponds to and note where these events
    % correlate to detections:
    for runs = 1:cycles
        n = rand;
        i = 1;
        while true
            if n<p(i+1)
                ys(runs) = event(i,1);
                xs(runs) = event(i,2);
                ws(runs) = event(i,3);
                zs(runs) = event(i,4);
                break
            end
            i=i+1;
        end
    end
    %% Dark Counts:
    % Scale percentage by 10^6 so that small percentages can affect it.
    thresh = dark_rate*1E8+.1;
    darkYs = randi(1E8,cycles,1)<thresh;
    darkXs = randi(1E8,cycles,1)<thresh;
    darkWs = randi(1E8,cycles,1)<thresh;
    darkZs = randi(1E8,cycles,1)<thresh;
    detectedYs = ys + darkYs;
    detectedXs = xs + darkXs;
    detectedWs = ws + darkWs;
    detectedZs = zs + darkZs;

    %% Place detected particles at a position:
    % Map particles to a location in it's quadrant with random azimuthal
    % and zenith angle, adding in a gaussian blur.
    sigma = blur;
    colist_test = cell(1,cycles);

    for i = 1:cycles
        cordlist_temp = [];
        theta = pi*rand;
        azim = pi*rand;
        xco = sin(theta)*cos(azim);
        yco = sin(theta)*sin(azim);
        zco = cos(theta);
        for j=1:detectedYs(i)
```

```
            cordlist_temp = [cordlist_temp; [xco yco 1+zco] + sigma*randn(1,3)];
        end
        for j=1:detectedXs(i)
            cordlist_temp = [cordlist_temp; [xco yco (-1+zco)] + sigma*randn(1,3)];
        end
        for j=1:detectedWs(i)
            cordlist_temp = [cordlist_temp; [-xco -yco (1-zco)] + sigma*randn(1,3)];
        end
        for j=1:detectedZs(i)
            cordlist_temp = [cordlist_temp; [-xco -yco (-1-zco)] + sigma*randn(1,3)];
        end
        %% Quantum Efficiency:
        % Cull data, keeping only the efficiency rating:
        thresh = quantum_efficiency +.01;
        keep_list = randi(100,size(cordlist_temp,1),1)<thresh;
        colist_test{i} = cordlist_temp.*keep_list;
    end
end
```