```
# John Wu
clear all;
clear drawRR;
clear drawLinks;

SetupRRParams();
delta_t = .02;
C_circ = [10 5]';
circ_rad = 5;
lambda = 1; %1 for ccw -1 for cw

A_pts = [12.5 0; 10 10; 7.5 0; 8.75 5; 11.25 5];

do_circ = 1;  %otherwise (0) draw square.


%Set up the initial conditions
if (do_circ) % 1 for circle, 0 for square
    config_deg = [45, -45]';  %beginning configuration - set to test
else
    config_deg = [51.3178, -51.3178*2]';
end
%% draw RR
%plot the robot arm and some imporant places
drawRR(config_deg, P);
if(do_circ)
    plot(C_circ(1), C_circ(2), 'r*', 'MarkerSize', 5);
else
    plot(A_pts(:,1), A_pts(:,2), 'r.', 'MarkerSize', 5);
end
omega = [0 0];  %sample omega control
Goal_Loc = [0, 0]';
wp = 1;

for t = 0:delta_t:15

    %Remember X_dot = J * THETA_dot
    pose_endEffector = forwardKinematics(config_deg, P);
    J = calculateJacobian(config_deg, P);
    J_inv = pinv(J(1:2,:));

    %Set the components of the workspace speed
    v = 8;  %linear speed in cm/s

    if(do_circ)
        %TODO Impliment code to follow a circle with radius 5 set at the center
        %point C_circ
        kd = .75;
        d = sqrt((pose_endEffector(1)-C_circ(1))^2+(pose_endEffector(2)-C_circ(2))^2)-↙
circ_rad;
        thetaCB = atan2d(pose_endEffector(2) - C_circ(2), pose_endEffector(1) - C_circ↙
```

```
(1));
        theta_c = thetaCB + lambda * (90+atand(kd*d));
        w = (theta_c);
        %if (w > pi)
        %    w = w - 2*pi;
        %elseif (w < -pi)
        %    w = w + 2*pi;
        %end
        w = w*1;
        X_dot = [v*cosd(w), v*sind(w)];
        omega = J_inv*X_dot';
    else
        %TODO impliment code to draw the line segments specified by the matrix
        %A_pts, break out of the loop when done.
        if(wp <= size(A_pts, 1))
            %Calculation for theta1_E
            Goal_Loc = [A_pts(wp, 1), A_pts(wp, 2)]';
            delta_pos = Goal_Loc - pose_endEffector(1:2);
            disToGoal = norm(delta_pos);
            if(disToGoal < .1)
                wp = wp + 1;
            else
                theta = atan2d(Goal_Loc(2) - pose_endEffector(2), Goal_Loc(1) -↙
pose_endEffector(1));
                X_dot = [v*cosd(theta), v*sind(theta)];
                omega = J_inv*X_dot';
            end
        else
            break
        end

    end


    %End TODO if you put your motor speeds in the omega row vector [omega(1), omega(2)]
    config_next = config_deg + rad2deg(delta_t*omega')';
    config_deg = config_next;
    drawRR(config_deg, P);

    pause(delta_t);
end
```