```
function [ state ] = WaypointController_roundCorner( state, W, P, circDist, WP )
%WaypointController_roundPoint

%TODO - Develop the code to move from waypoint to waypoint inserting
%fillets into the waypoints to enable rounded corners.

%Utilize your line and circle following functions from the previous projects.

    % Called from WaypointController_throughPoint
    % When done, falls through and returns state to throughPoint
    vs = P.v_const * .8; % Sets velocity set point to .8 of the max
    circleRad = P.wheel_base/(2*(P.v_const/vs-1));
    if(WP < size(W,2))
        vecNext = [W(1, WP+1) - W(1, WP); W(2, WP+1) - W(2, WP)];
    end
    if(WP ~= 1)
        vecLast = [W(1, WP-1) - W(1, WP); W(2, WP-1) - W(2, WP)];
    else
        vecLast = [0 - W(1, WP); 0 - W(2, WP)];
    end
    nextHat = vecNext/norm(vecNext);
    lastHat = vecLast/norm(vecLast);
    circHat = (vecNext + vecLast)/norm(vecNext + vecLast);
    halfangle = atan2(circHat(2), circHat(1));
    lambda = (-lastHat(1)*nextHat(2) + lastHat(2)*nextHat(1));
    lambda = (lambda/abs(lambda));
    % Uses the halfangle (of the angle between waypoints) to find where the
    % circle should be centered
    circx = W(1, WP) + cos(halfangle)*circDist;
    circy = W(2, WP) + sin(halfangle)*circDist;
    circleToFollow = [circx; circy; circleRad];
    drawCircle(circleToFollow, 'r');

    while(sqrt((W(1,WP)-state(1))^2+(W(2,WP)-state(2))^2) <= circDist)
        state = FollowCircle(state, circleToFollow, lambda, P.delta_t, P);
    end

end
```