```
function [ state ] = WaypointController_throughPoint( state, W, P )
%WaypointController

%TODO - Develop the code to move from waypoint to waypoint going through
% the waypoint (or the halfplane at the waypoint) before moving to the next
%Utilize your line following functions from the previous projects.
    for(i=1:size(W,2)) % go through waypoints
        % Path to waypoint, calculate halfpoint each time
        % Make halfpoint the center of the circle.
        % If halfpoint reached, go to round corner
        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(i == 1)
            angle = atan2(W(2,1)-0, W(1,1)-0);
        else
            angle = atan2(W(2,i)-W(2,i-1), W(1,i)-W(1,i-1));
        end
        % Takes angle at corner to calculate circle distance
        if(i < size(W,2))
            vecNext = [W(1, i+1) - W(1, i); W(2, i+1) - W(2, i)];
        end
        if(i ~= 1)
            vecLast = [W(1, i-1) - W(1, i); W(2, i-1) - W(2, i)];
        else
            vecLast = [0 - W(1, i); 0 - W(2, i)];
        end
        circHat = (vecNext + vecLast)/norm(vecNext + vecLast);
        halfangle = atan2(circHat(2), circHat(1));
        if(i == size(W,2))
            circDist = 30;
        else
            circDist = abs(circleRad/sin(halfangle));
        end
        % Takes angle and gives it with respect to the positive x axis
        if(i == 1)
            lastx = 0;
            lasty = 0;
        else
            lastx = W(1, i-1);
            lasty = W(2, i-1);
        end
        % Creates a new line to follow for each waypoint
        lineToFollow = [W(1, i); W(2, i); angle];
        % Sees how far from the destination we are
        destDistance = sqrt((W(1,i)-state(1))^2+(W(2,i)-state(2))^2);
        % We're close enough if we're within a radius of the circle's
        % distance
        while(destDistance > circDist)
            state = FollowLine(state, lineToFollow, P.delta_t, P);
            destDistance = sqrt((W(1,i)-state(1))^2+(W(2,i)-state(2))^2);
        end
```

```
        % Close enough to the corner, let's round the corner.
        if(i < size(W, 2))
            state = WaypointController_roundCorner(state, W, P, circDist, i);
        end
    end

end
```